XDef 2019

对话·交流·合作　前沿·实用·人才

第八届全国网络与信息安全防护峰会

CPU芯片硬件安全系统解决方案

刘雷波

清华大学微电子学研究所

提 纲

一、技术背景

二、我国现有应对方案

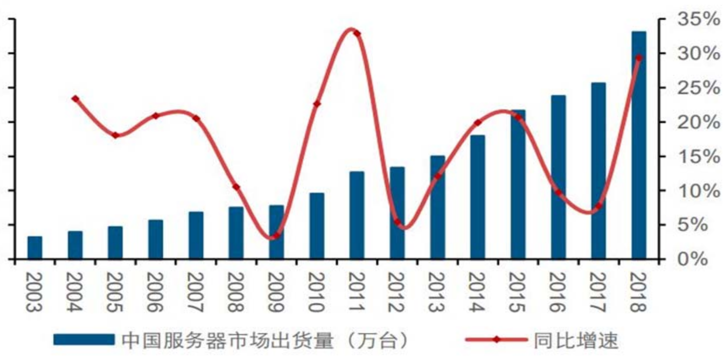
三、CPU硬件安全动态监测管控技术

四、技术应用探讨

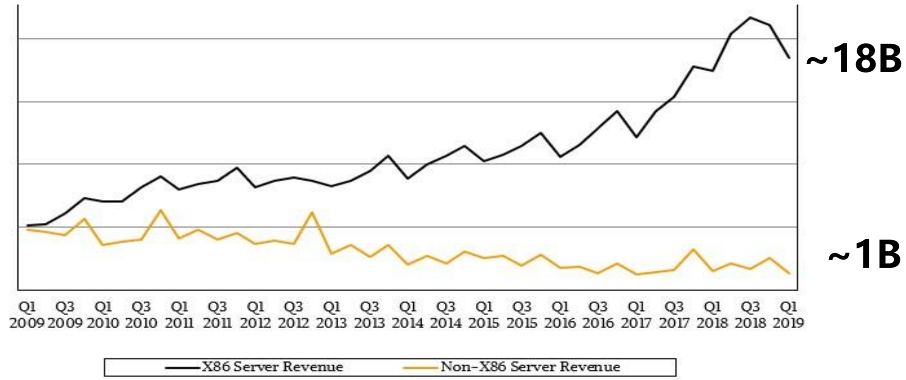
五、总结

一、技术背景 - 我国通用服务器CPU现状

- 我国通用服务器CPU
  - **存量>1500万台，年均增量~250万台 （近4年）**
  - **>95%为X86**，X86中>99%为 Intel 至强（Xeon）
- 全球通用服务器CPU >95%为X86

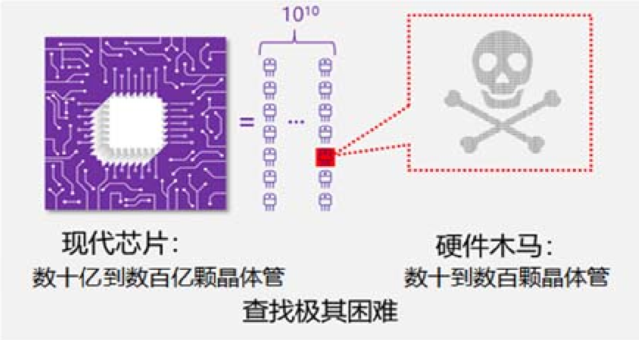


2003-2018中国服务器出货量, IDC, 2019



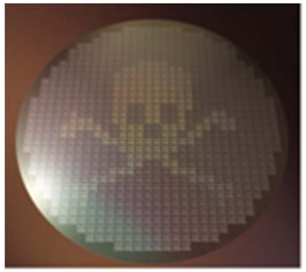
2009-2019全球X86与非X86服务器销售额, IDC, 2019

一、技术背景 - 芯片硬件安全问题

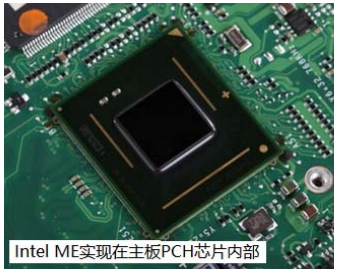




- 恶意硬件（木马/后门）、硬件漏洞、硬件前门等
  - **尚无有效手段能够进行检测和抵御**







一、技术背景 - CPU芯片已知硬件安全问题

- **硬件漏洞**
  - 熔断（Meltdown）[1]、幽灵（Spectre）[2]及其变种
  - 其他软件侧信道攻击，如Branch Scope[3], TLBleed[4] , VolJokey[5] , ……
- **硬件前门（被利用后可能成为恶意硬件）**
  - 处理器微码，如Intel，AMD
  - 主动管理子系统，如Intel ME，AMD PSP
- **恶意硬件（木马/后门）**
  - 有相关报道[6]，尚无我们可复现的实例

[1] Meltdown, https://meltdownattack.com/
[2] Spectre Attacks..., https://meltdownattack.com/
[3] BranchScope: A New..., ASPLOS’ 18, March 24–28, 2018
[4] Translation Leak-aside..., USENIX Security Symposium, 2018
[5] CCS 2019
[6] Breakthrough..., CHES 2012.14th international workshop, 2012

一、技术背景 - 硬件漏洞（熔断、幽灵及其变种）



云服务导致企业重要数据泄露



内网普通用户获取高权限数据

SPECTRE

MELTDOWN



手机端个人隐私泄露

一、技术背景 - 熔断、幽灵原理分析

- **由CPU技术原理引发的硬件漏洞**
  - 非程序员的疏忽

	• 乱序执行	• 执行结果
1. ; rcx = kernel address 2. ; rbx = probe array 3. retry: 4. mov al, byte [rcx]		指令提交将产生异常
5. shl rax, 0xc 6. jz retry 7. mov rbx, qword [rbx + rax]	异常被执行前，line 7 有机会完成访存，从而改变Cache布局	以下指令不会被提交 CPU将转入异常处理
	微结构状态改变	体系结构状态不变

熔断攻击（概念原型代码）

	• 推测执行	• 执行结果
• x先赋值为较小数，反复训练分支预测器 • 使用特定操作，令条件判断执行时延变长 • 保证probe数组被清除出Cache • 保证目标数据已加载在Cache中 • x赋值为大数，令array[x] 指向目标数据 1. if (x < array_size)	预测条件为真	判断语句提交，条件为假
2. y = probe[array[x] * 256];	有机会执行访存，从而改变Cache布局	条件为假，不会提交
	微结构状态改变	体系结构状态不变

幽灵攻击（概念原型代码）

• 详解熔断攻击

		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0					DI	RF	ALU1	WB	CO								
1						DI	RF	ALU1	WB	CO							
2							D										
3																	
4	ad… rax		FE	DE	RN												
5	a… 1, %rax		FE	DE	RN												
6	…x141, %rax		FE	DE	RN						DI	RF	ALU1	WB	CO		
7	… $0x141, %rax		FE	DE	RN							DI	RF	ALU1	WB	CO	
8	movb (%rcx), %al			FE	DE	RN	DI	RF	LSU	LSU	WB						CO
9	shl $12, %rax			FE	DE	RN				DI	RF	ALU…					
10	jz retry			FE	DE	RN					DI	RF	ALU2	WB			
11	add $target, …			FE	DE	RN						DI	RF	ALU2	WB		
12	movq (%rax)…				FE	DE	RN						DI	RF	LSU	LSU	LSU

异常延迟提交

FE:取指令
DE:指令解码
RN:寄存器重命名

③ 指令9，10，11，12得以使用保留站中存放的指令8的结果，继续执行

⑤ 发生异常的指令8提交，提交后将触发CPU跳入异常处理，并冲刷流水线使得后续的指令无法提交。

① 指令8（movb (%rcx), %al）是一条越权读取内核数据的指令

② 权限检查失败， 由于CPU使用精确异常 (CPU需要知道哪条条指令发生了异常），并不立即FLUSH流水线，而是在reorder buffer对应的指令8位置标记一个异常。由于指令0-7使用和指令8相同的目的寄存器，存在写后写依赖，指令8只能在指令7提交后的周期15提交

④ 权限检查虽然失败，但在此之前在周期13，利用内核数据作为索引的访存请求已经发送给内存子系统，导致这些数据从内存加载到了cache

一、技术背景 - 熔断、幽灵原理分析

- **由CPU技术原理引发的硬件漏洞**
  - 非程序员的疏忽
- **软件无足够信息进行有效甄别及防范**
  - 软件不了解CPU部分重要硬件行为

	• 乱序执行	• 执行结果
1. ; rcx = kernel address 2. ; rbx = probe array 3. retry: 4. mov al, byte [rcx]		指令提交将产生异常
5. shl rax, 0xc 6. jz retry 7. mov rbx, qword [rbx + rax]	异常被执行前，line 7 有机会完成访存，从而改变Cache布局	以下指令不会被提交 CPU将转入异常处理
	微结构状态改变	体系结构状态不变

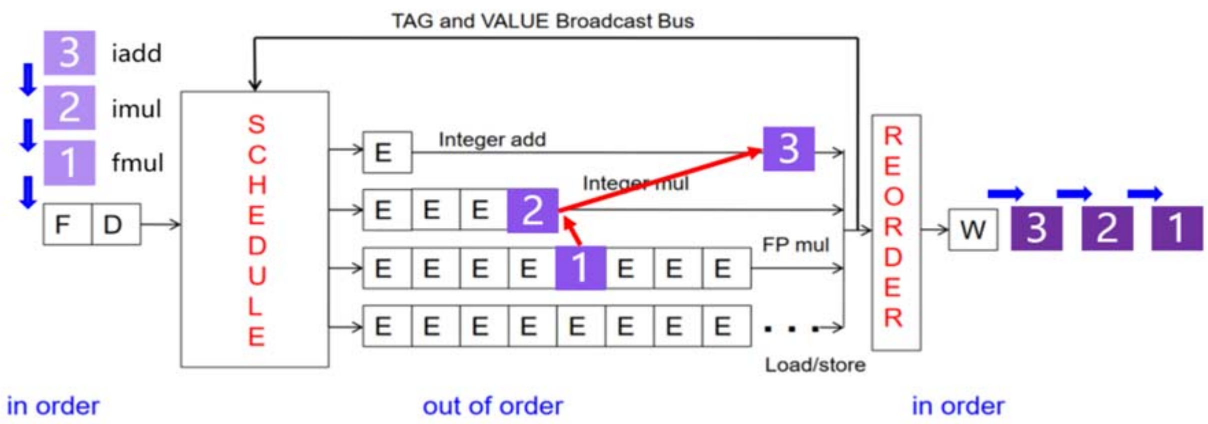
熔断攻击（概念原型代码）

	• 推测执行	• 执行结果
• x先赋值为较小数，反复训练分支预测器 • 使用特定操作，令条件判断执行时延变长 • 保证probe数组被清除出Cache • 保证目标数据已加载在Cache中 • x赋值为大数，令array[x] 指向目标数据 1. if (x < array_size)	预测条件为真	判断语句提交，条件为假
2. y = probe[array[x] * 256];	有机会执行访存，从而改变Cache布局	条件为假，不会提交
	微结构状态改变	体系结构状态不变

幽灵攻击（概念原型代码）

一、技术背景 - 熔断、幽灵原理分析

软件可见 | **软件不可见** | 软件可见



处理器乱序执行示意

一、技术背景 - 熔断、幽灵原理分析

- **由CPU技术原理引发的硬件漏洞**
  - 非程序员的疏忽
- **软件无足够信息进行有效甄别及防范**
  - 软件不了解CPU部分重要硬件行为
- **解决问题需彻底改变现有CPU架构**
  - CPU性能损失无法接受
    - 禁用推测/乱序等技术
    - 对推测执行/乱序等进行权限检查
    - 重新设计Cache

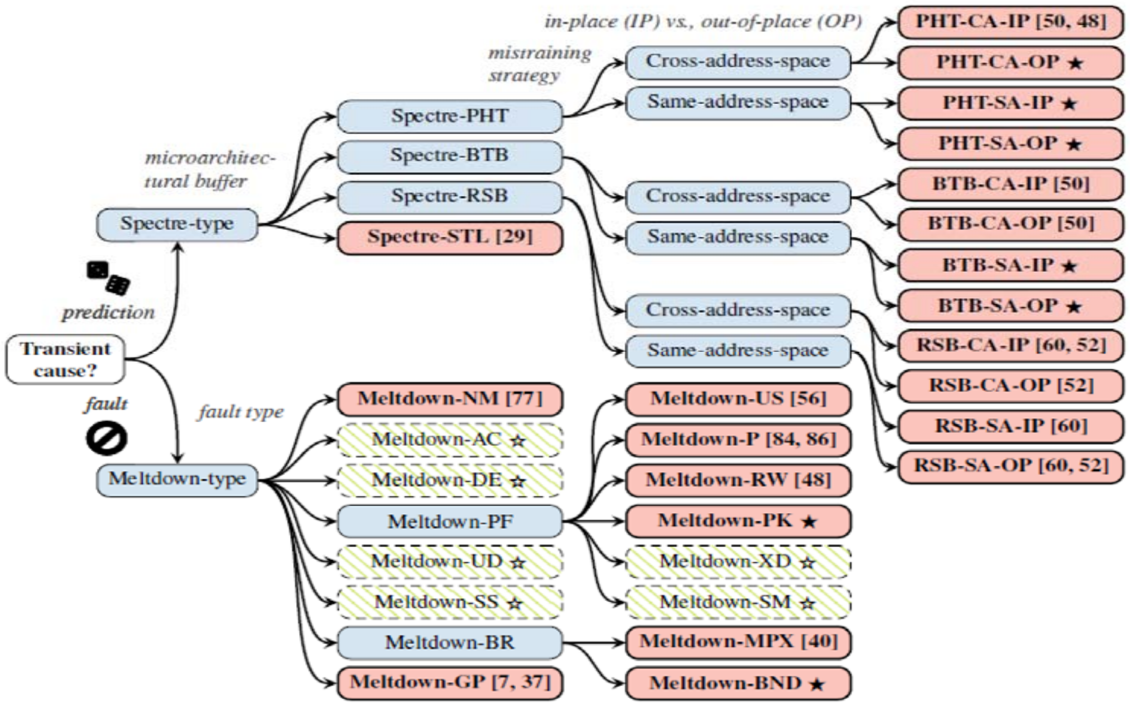


熔断攻击（概念原型代码）



幽灵攻击（概念原型代码）

一、技术背景 - 熔断、幽灵的变种及其持续演进[1]



[1] Gruss, et al. A Systematic Evaluation of Transient Execution Attacks and Defenses. 28th USENIX Security Symposium 2019.

一、技术背景 - 硬件漏洞（非熔断、幽灵攻击原理）

- Branch Scope（ASPLOS 2018）
  - **旁路攻击CPU的分支预测器**
  - 泄露SGX (软件保护扩展）防护的数据
- TLBleed（USENIX Security 2018）
  - **旁路攻击CPU的TLB(地址翻译缓冲)**
  - 破坏进程间隔离，造成数据泄露
- VolJokey（CCS 2019）
  - **旁路攻击CPU的驱动电压**
  - 泄露重要数据

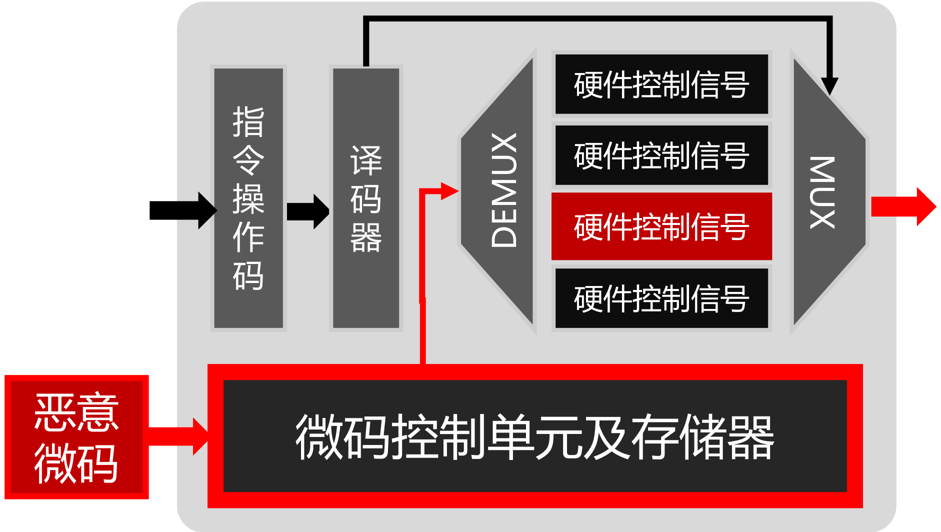
一、技术背景 - CPU芯片已知硬件安全问题

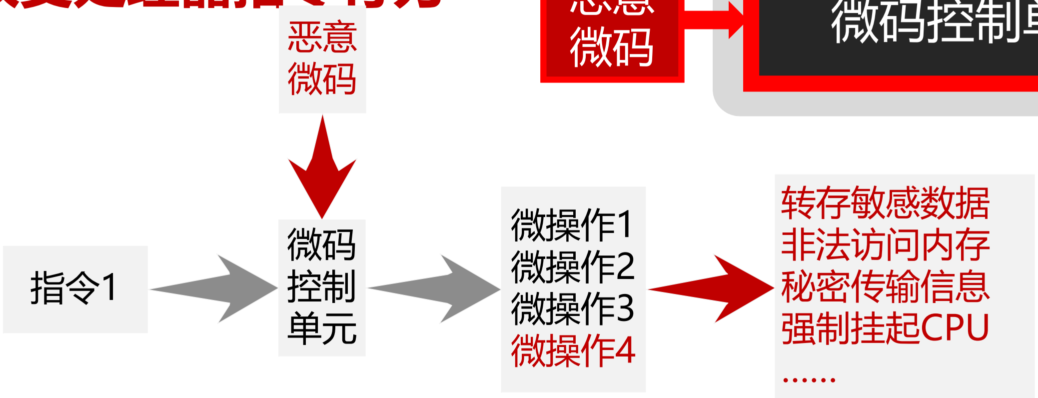
- 硬件漏洞
  - 熔断（Meltdown）[1]、幽灵（Spectre）[2]及其变种
  - 其他软件侧信道攻击，如Branch Scope[3], TLBleed[4] , VolJokey[5] , ……
- **硬件前门（被利用后可能成为恶意硬件）**
  - 处理器微码，如Intel，AMD
  - 主动管理子系统，如Intel ME，AMD PSP
- 恶意硬件（木马/后门）
  - 有相关报道[6]，尚无我们可复现的实例

[1] Meltdown, https://meltdownattack.com/
[2] Spectre Attacks..., https://meltdownattack.com/
[3] BranchScope: A New..., ASPLOS’18, March 24–28, 2018
[4] Translation Leak-aside..., USENIX Security Symposium, 2018
[5] CCS 2019
[6] Breakthrough..., CHES 2012.14th international workshop, 2012

一、技术背景 - 硬件前门（处理器微码）

- 微码（Intel、AMD）
  - **厂商预留，不公开**
    - 用于硬件维护与升级
    - 性能提升的重要手段
  - **可改变处理器指令行为**



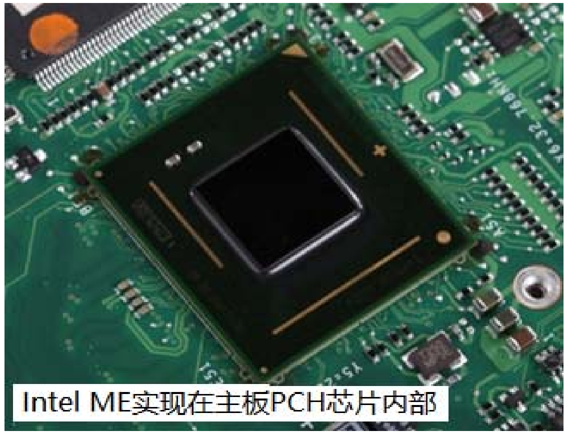


一、技术背景 - 硬件前门（主动管理子系统）

- 主动管理子系统（ME、PSP）
  - **厂商预留、不公开**
  - 绕过CPU，直接访问内存资源



可通过ME攻击一台已关闭的电脑： 完成启动，
远程控制等一系列操作



主板PCH芯片（ME内嵌其中）

- 子系统前门的危害
  - **电脑关闭，仍可被远程控制**
  - **软件手段无法察觉**

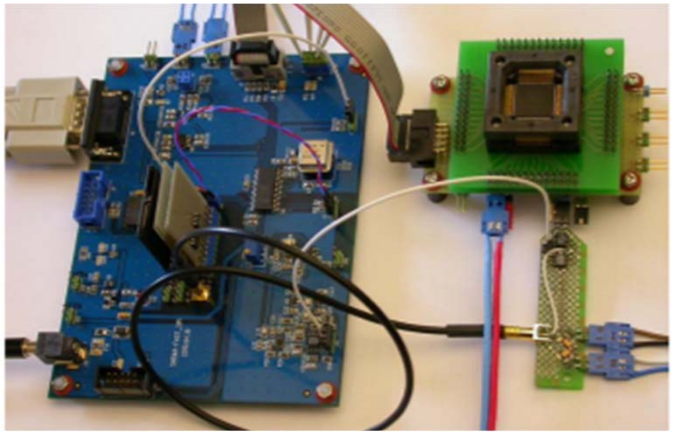
一、技术背景 - CPU芯片已知硬件安全问题

- **硬件漏洞**
  - 熔断（Meltdown）[1]、幽灵（Spectre）[2]及其变种
  - 其他软件侧信道攻击，如Branch Scope[3], TLBleed[4] , VolJokey[5] , ...
- **硬件前门（被利用后可能成为恶意硬件）**
  - 处理器微码，如Intel，AMD
  - 主动管理子系统，如Intel ME，AMD PSP
- **恶意硬件（木马/后门）**
  - 有相关报道[6]，**尚无我们可复现的实例**

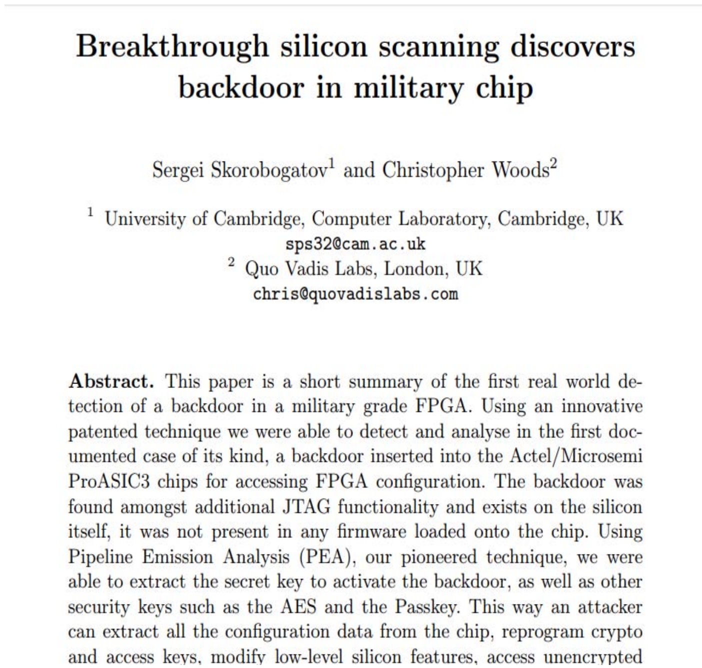
[1] Meltdown, https://meltdownattack.com/
[2] Spectre Attacks..., https://meltdownattack.com/
[3] BranchScope: A New..., ASPLOS’18, March 24–28, 2018
[4] Translation Leak-aside..., USENIX Security Symposium, 2018
[5] CCS 2019
[6] Breakthrough..., CHES 2012.14th international workshop, 2012

一、技术背景 - 恶意硬件（木马、后门）

- **首个公开报道的真实芯片中的木马**
  - 军用FPGA芯片
- 与密码运算相关的重要数据遭泄露



恶意硬件原型验证板



Breakthrough silicon scanning discovers backdoor in military chip

Sergei Skorobogatov[1] and Christopher Woods[2]

[1] University of Cambridge, Computer Laboratory, Cambridge, UK
sps32@cam.ac.uk
[2] Quo Vadis Labs, London, UK
chris@quovadislabs.com

Abstract. This paper is a short summary of the first real world detection of a backdoor in a military grade FPGA. Using an innovative patented technique we were able to detect and analyse in the first documented case of its kind, a backdoor inserted into the Actel/Microsemi ProASIC3 chips for accessing FPGA configuration. The backdoor was found amongst additional JTAG functionality and exists on the silicon itself, it was not present in any firmware loaded onto the chip. Using Pipeline Emission Analysis (PEA), our pioneered technique, we were able to extract the secret key to activate the backdoor, as well as other security keys such as the AES and the Passkey. This way an attacker can extract all the configuration data from the chip, reprogram crypto and access keys, modify low-level silicon features, access unencrypted

2012年，剑桥大学[1]

[1] Breakthrough silicon scanning discovers backdoor in military chip, CHES 2012

一、技术背景 - 总结

- **我国服务器CPU芯片在较长一段时期内主要依赖进口**
- 进口服务器CPU芯片的典型安全特征

1、黑盒（封装好的芯片或管芯）

- 使用者不掌握实现细节
- 使用者不掌握检测手段

2、存有硬件安全隐患

- 严重程度未知
- 大多数攻击发生在商业部署后
- 软件手段无法单独解决



提 纲

一、技术背景

二、我国现有应对方案

三、CPU硬件安全动态监测管控技术

四、技术应用探讨

五、总结

二、我国现有应对方案 - 软件安全与硬件安全的边界

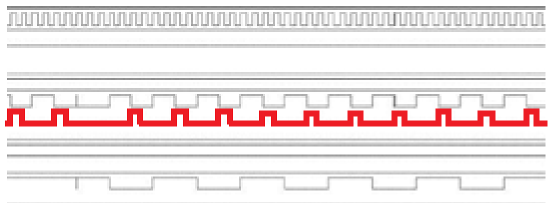
- 软件安全

1 ldm r1-r15,(r0) 2 add r0, r1 3 addi r0, 28	1 **addi r0, 32** 2 **ldm r1-r15,(r0)** 3 addi r0, 28	1 ldm r1-r15,(r0) 2 addi r0, **36** 3 addi r0, 28
指令本身被篡改	**执行执行顺序被篡改**	**指令执行环境被篡改**，如寄存器、存储器、立即数被篡改

- **硬件安全：指令底层硬件行为被篡改，如输出信号被篡改**







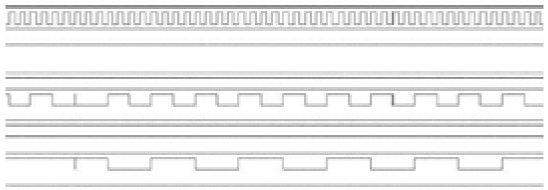
二、我国现有应对方案 - 软件安全与硬件安全的边界

1 操作码 操作数
2 操作码 操作数
3 操作码 操作数
4 操作码 操作数
5 操作码 操作数

正常指令

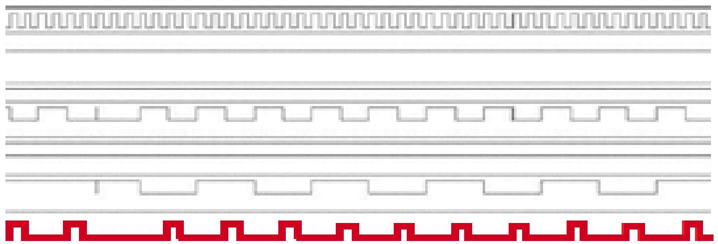


芯片内恶意硬件



CPU芯片输出信号：恶意硬件未被触发，输出信号正确，既定功能正确

CPU芯片输出信号：恶意硬件被触发，既定功能正确，但新增一系列非预期输出信号。



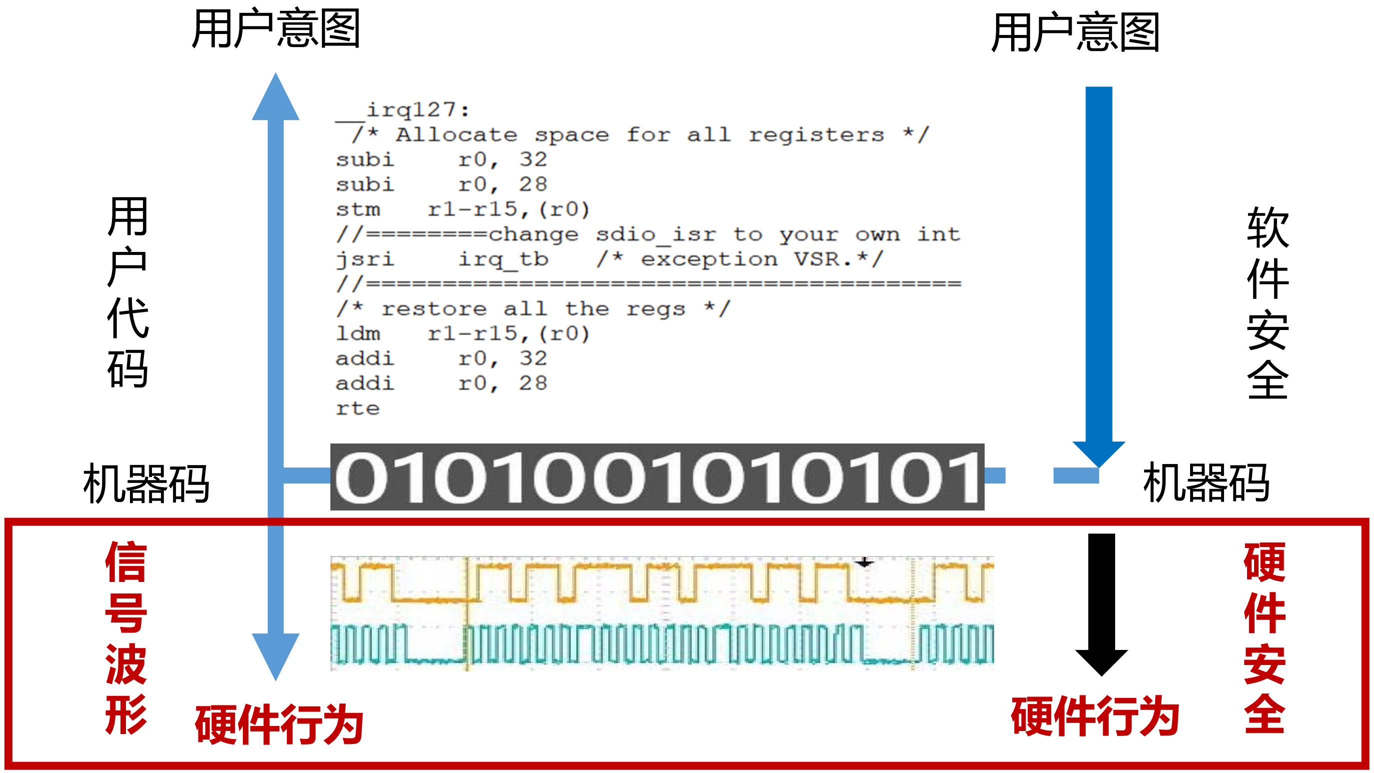
CPU芯片的功能测试、可靠性测试，及软件安全手段：无法查出该类问题。





…

二、我国现有应对方案 - 软件安全与硬件安全的边界



二、我国现有应对方案 - 两个需要回答的重要问题

- CPU芯片中物理信号的数量与变化空间极大，难以穷举
  - 是否符合预期的参照依据是什么?
  - 芯片功能测试、可靠性测试等手段能否甄别?

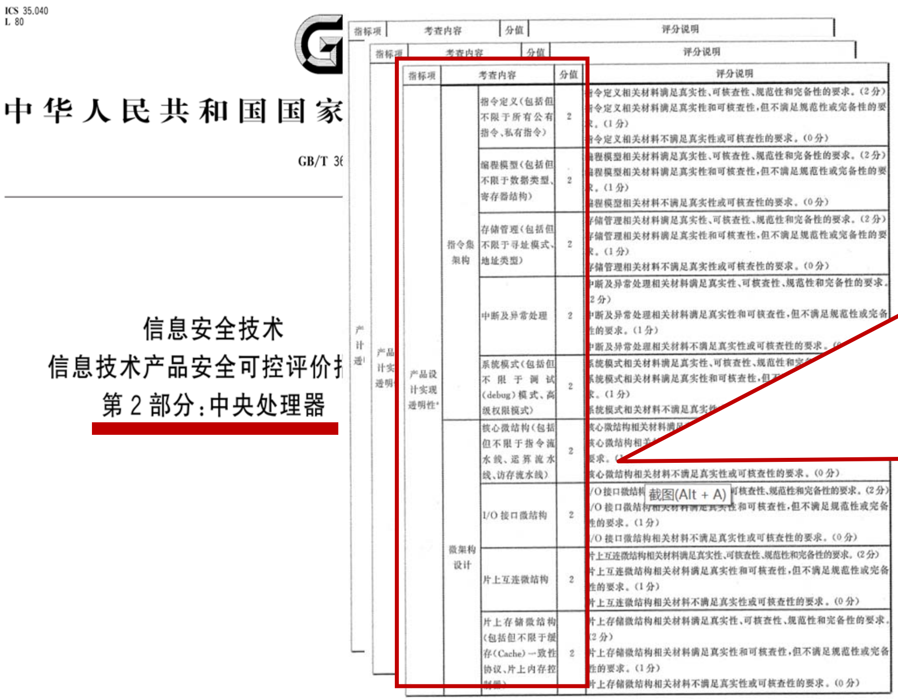
- CPU芯片中的物理信号符合预期，就不存在硬件安全问题吗?

二、我国现有应对方案

CPU商业部署前	CPU商业部署后
1、代码和流程审查 2、剖片检测，反向分析	3、固件验证（防篡改、来源验证等）

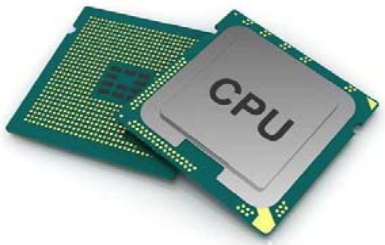
二、我国现有应对方案 - 部署前：代码和流程审查

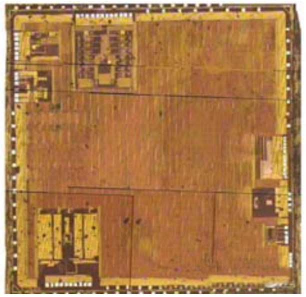
- 中央处理器安全可控评价指标（国家标准）2019.4.1正式实施

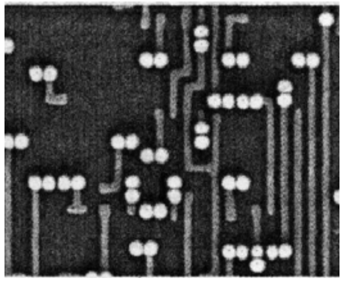


评价类别	细分考察项	分值
设计实现透明性	指令集架构、微架构设计、物理设计	28分
设计验证	验证环境、验证充分性、验证结果与产品一致性	25分
关键技术研发能力	指令集架构、关键IP核研发能力	14分
安全生态适应性	操作系统适配能力、密码合规性	8分
持续供应能力	产品供应情况、核心团队情况	7分
供应链保障能力	设计、流片、封装、测试环节	10分
服务保障能力	服务及时性、规范性	8分

二、我国现有应对方案 - 部署前检测：剖片/逆向分析[1][2]







芯片 → 管芯照片 → 电路

问题：① 对硬件安全威胁的先验知识非常有限

② 工程上难以实现

[1] 集成电路芯片安全隐患检测技术，半导体技术第32卷第12期，2007

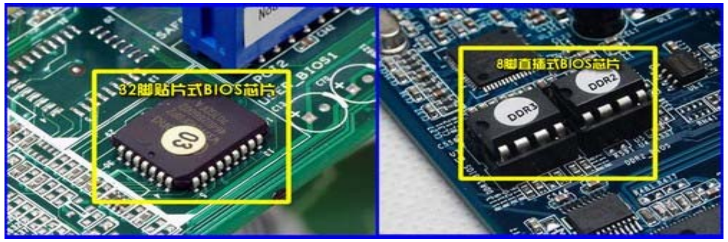
[2] 一种基于芯片解剖照片的IP核侵权鉴定方法，第十六届计算机工程与工艺年会暨第二届微处理器技术论坛论文集，2012

二、我国现有应对方案

CPU商业部署前	CPU商业部署后
1、代码和流程审查 2、剖片检测，反向分析	**3、固件验证（防篡改、来源验证等）**

二、我国现有应对方案 - 部署后方法：固件验证

- CPU启动时**对固件进行防篡改验证**，如对BIOS进行校验



- CPU启动时**对固件进行来源验证**，如对固件证书进行验签

证书签发
服务器



签发证书



固件验证



- 问题： 固件只是硬件安全威胁的冰山一角，**硬件攻击很可能是在CPU正常运行时发生。**

二、我国现有应对方案 - 总结

现有方法远远无法解决已知硬件安全问题

- 黑名单机制 → 恶意攻击检测的伪阴性
- 非运行时检测 → 功能测试的伪阴性

CPU商业部署前	CPU商业部署后
1、代码和流程审查 2、剖片检测，反向分析	3、固件验证（防篡改、来源验证等）

提 纲

一、技术背景

二、我国现有应对方案

三、CPU硬件安全动态监测管控技术 （DSC）

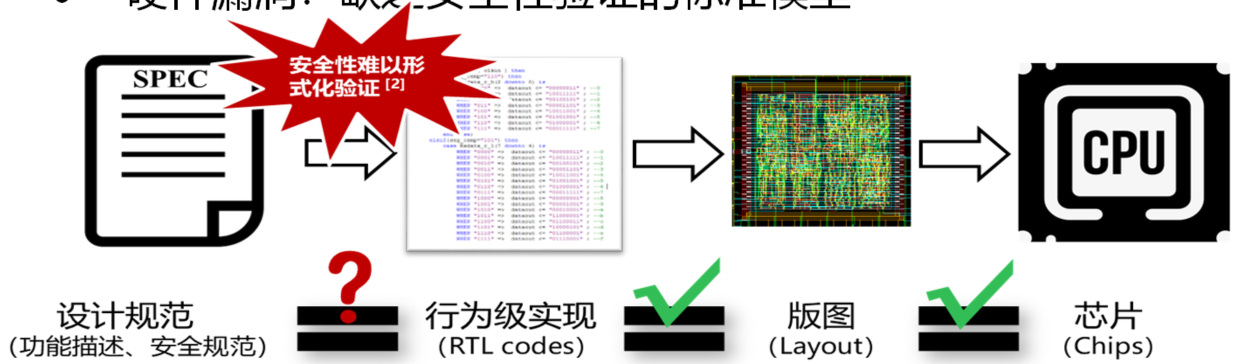
四、技术应用探讨

五、总结

三、CPU硬件安全动态监测管控技术

CPU硬件安全问题难以完全解决的本质原因

- **CPU硬件实现与CPU设计规范的安全一致性目前是不可验证的[1]**
  - 恶意硬件：安全性验证的测试空间过大
  - 硬件漏洞：缺乏安全性验证的标准模型



[1] S. Bhunia, et al, "Hardware Trojan attacks: threat analysis and countermeasures," Proceedings of the IEEE, 2014.
[2] C. Sturton, "FinalFilter: Asserting Security Properties of a Processor at Runtime," IEEE Micro, 2019.

三、CPU硬件安全动态监测管控技术

“理论情况下，可验证正确与安全的CPU是可实现的。然而，现在CPU的复杂度使这完全不可实现。**完整验证一个现代CPU仍然是非常棘手的。即使只是静态地验证一个非常简单的安全属性，如‘禁止硬件权限提升’，对于当前技术而言都是无法做到的”[1]。**

——J. M. Smith, DARPA



Expert Opinion

FinalFilter: Asserting Security Properties of a Processor at Runtime

Cynthia Sturton
University of North Carolina at Chapel Hill

Matthew Hicks
Virginia Tech

Samuel T. King
University of California, Davis

Jonathan M. Smith
University of Pennsylvania and DARPA

In an ideal world, it would be possible to build a provably correct and secure processor. However, the complexity of today's processors puts this ideal out of reach. The complete verification of a modern processor remains intractable. Statically verifying even a simple security property—for example, "*hardware privilege escalation never occurs*"—remains beyond the state of the art in formal verification.

Testing can complement formal verification methods, yet testing is incomplete and bugs in the hardware that leave it vulnerable continue to elude test suites. Further, a crafty malicious actor can evade typical testing coverage metrics.

Recent efforts, including that of three of the authors, have explored the use of static analysis on the design files (e.g., hardware description level source code or gate-level netlists) to find suspicious circuitry.[1–3] These techniques rely on heuristics to define patterns that indicate a likely trojan and then search for instances in the design that match the pattern. However, malicious circuitry that does not match the pattern will be missed, as will inadvertent bugs that open vulnerabilities. By the time the weakness is uncovered, the hardware is already in the end user's hands and vulnerable to attack.

In the absence of a full proof of correctness, what is needed is a final filter: a runtime verification technique that works—postdeployment—to detect and respond to security property violations as they occur during execution. In this article, we make the case for final filters using our tool, FinalFilter, as a case study.

FINALFILTER

Prior research, including our own, has shown that assertions hard-coded into the design can be a cheap and effective way to verify the correctness of any single execution run.[4,5] Assertions can cover properties that would be intractable to prove statically for the current state of the art. The downside

Digital Object Identifier 10.1109/MM.2019.2921509
Date of current version 23 July 2019.

July/August 2019 Published by the IEEE Computer Society 0272-1732 © 2019 IEEE 35

[1] C. Sturton, "FinalFilter: Asserting Security Properties of a Processor at Runtime," IEEE Micro, 2019.

三、CPU硬件安全动态监测管控技术

3.1 技术原理

3.2 工作流程

3.3 关键技术

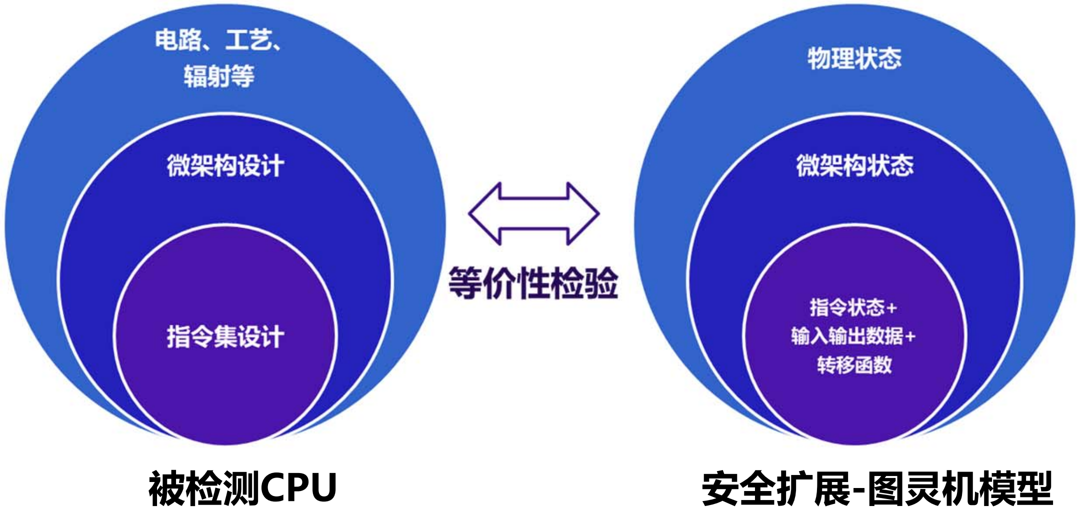
3.4 技术验证

3.1 技术原理 - CPU硬件安全动态监测管控技术

- CPU以图灵机模型为基础，所以：

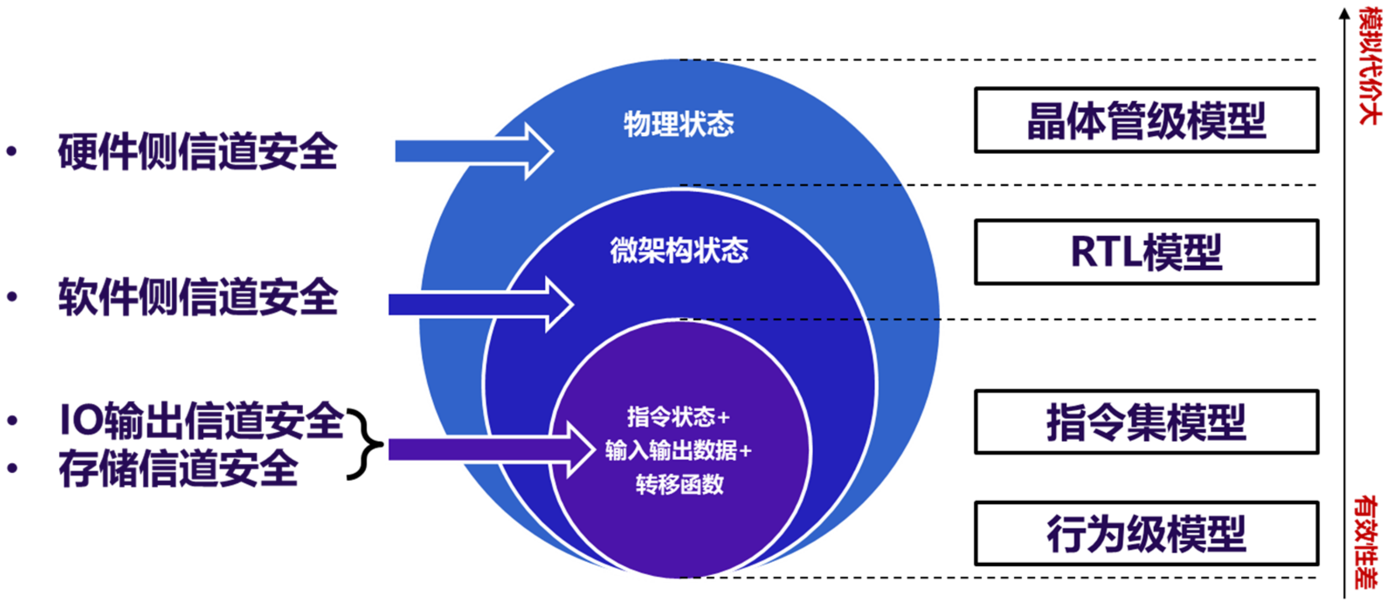
  （1）**CPU行为具有确定性**

  （2）**CPU行为安全可用等价的图灵机模型进行等价性检验**



3.1 技术原理 - CPU硬件安全动态监测管控技术

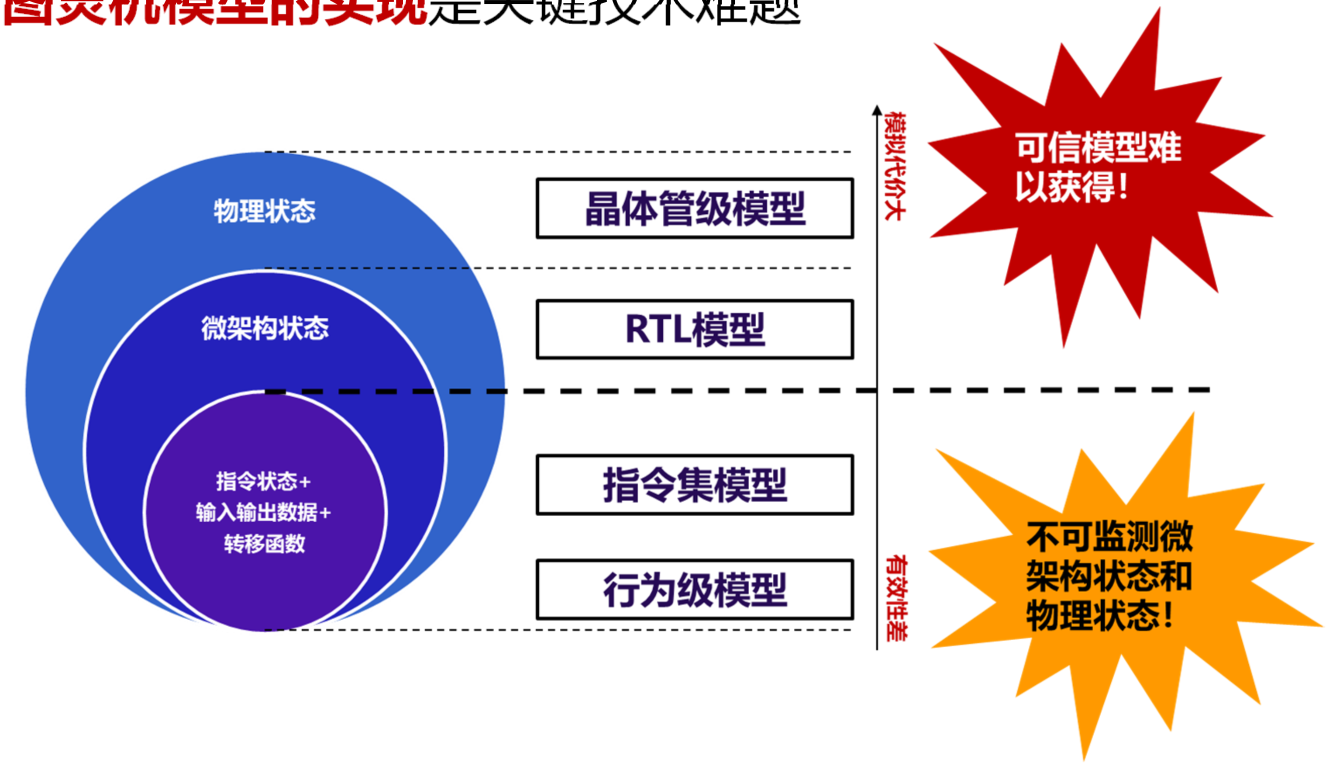
- **构建等价图灵机模型是实现CPU硬件安全监测的关键**



图灵机模型级别

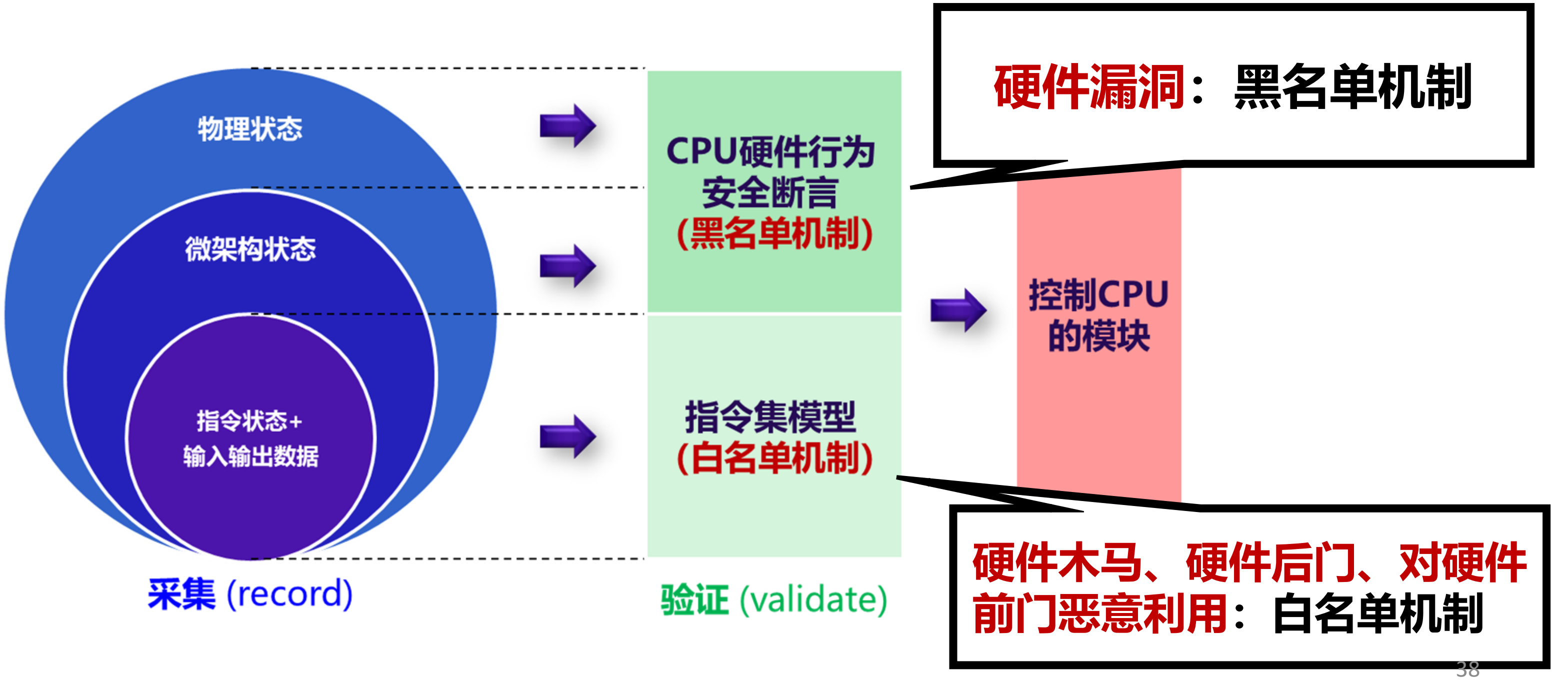
3.1 技术原理 - CPU硬件安全动态监测管控技术

- **等价图灵机模型的实现**是关键技术难题

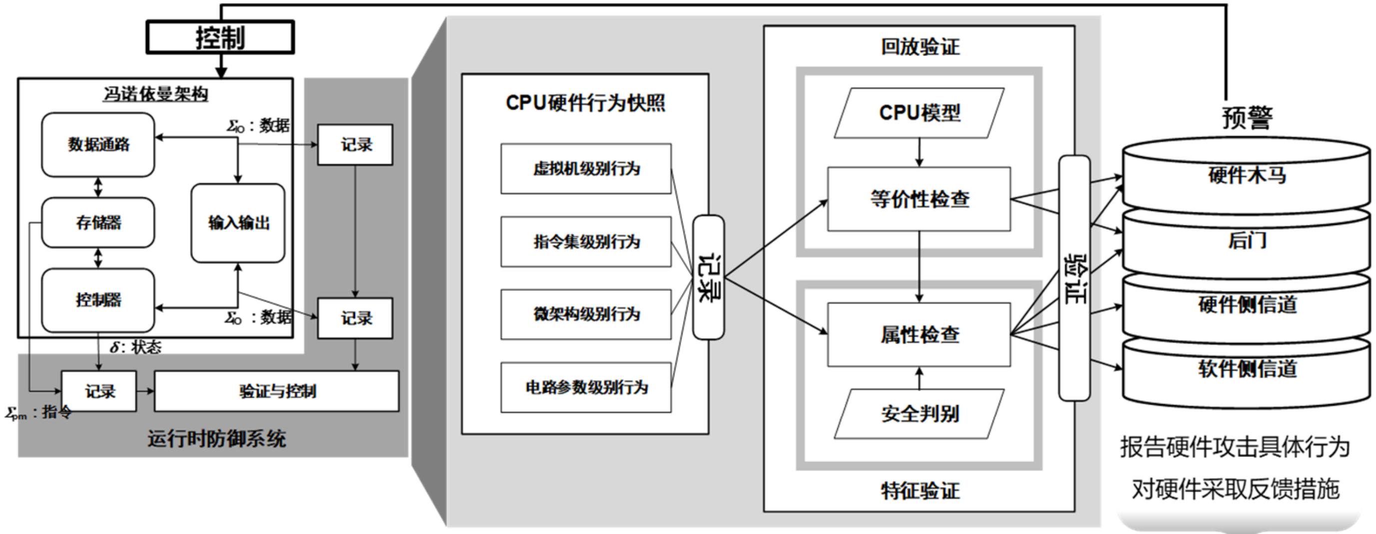


3.1 技术原理 - CPU硬件安全动态监测管控技术

- **“指令级CPU模型 + 硬件行为安全断言”**的动态监控技术



3.1 技术原理 - DSC技术原理示意图



采集 (record) **验证** (validate) **管控**(control)

有限长度行为样本 + 硬件状态迁移过程 + 指令级CPU模型+硬件行为安全断言

三、CPU硬件安全动态监测管控技术

3.1 技术原理

3.2 工作流程

3.3 关键技术

3.4 技术验证

3.2 工作流程 - CPU硬件安全动态监测管控技术实现框架

行为采集

- 采集CPU所有对外物理通信
- 采集CPU内部状态（CPU提供商允许访问的内部状态）

⇩

行为分析

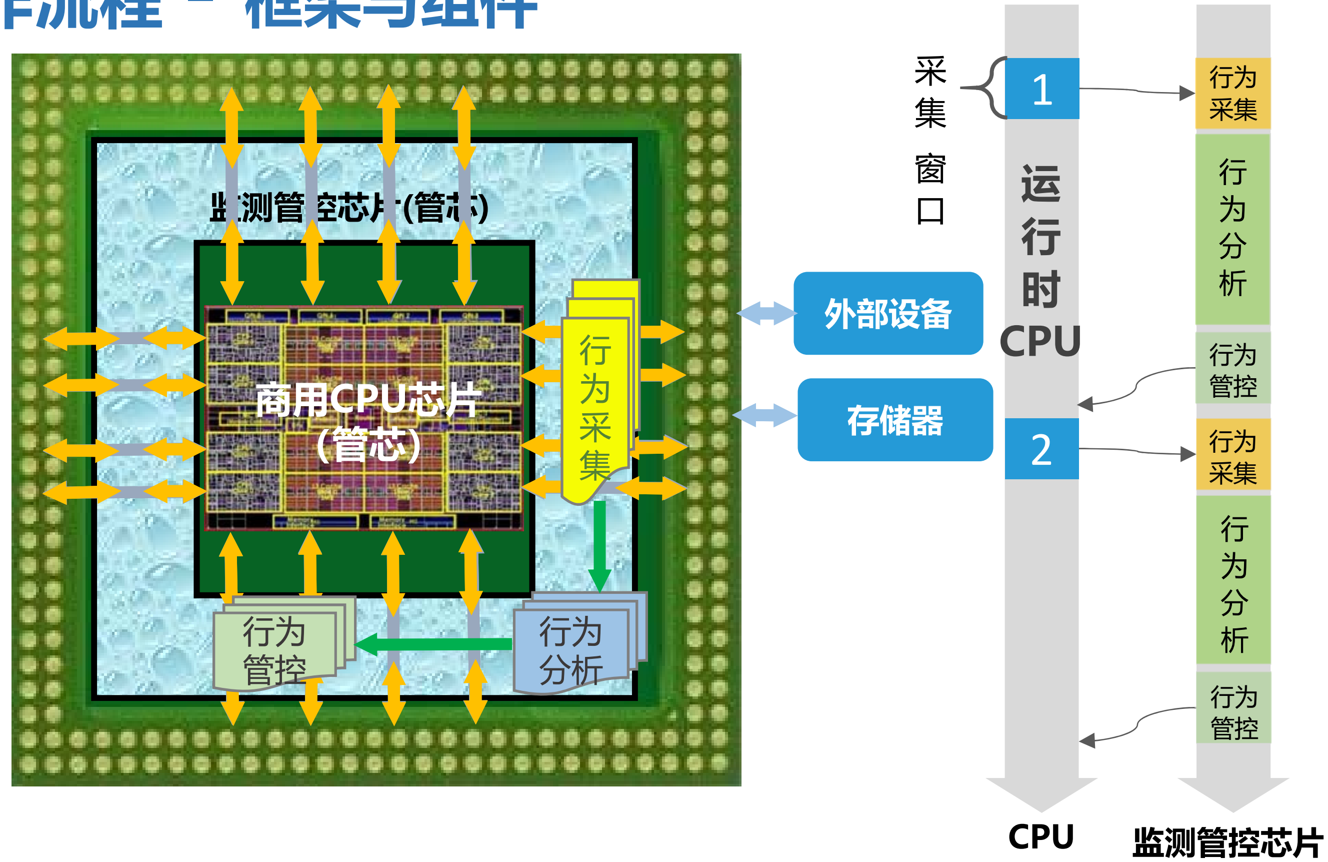
- 基于ISA模型对采集到的行为进行回放与分析
- 基于安全断言对采集到的行为进行判断

⇩

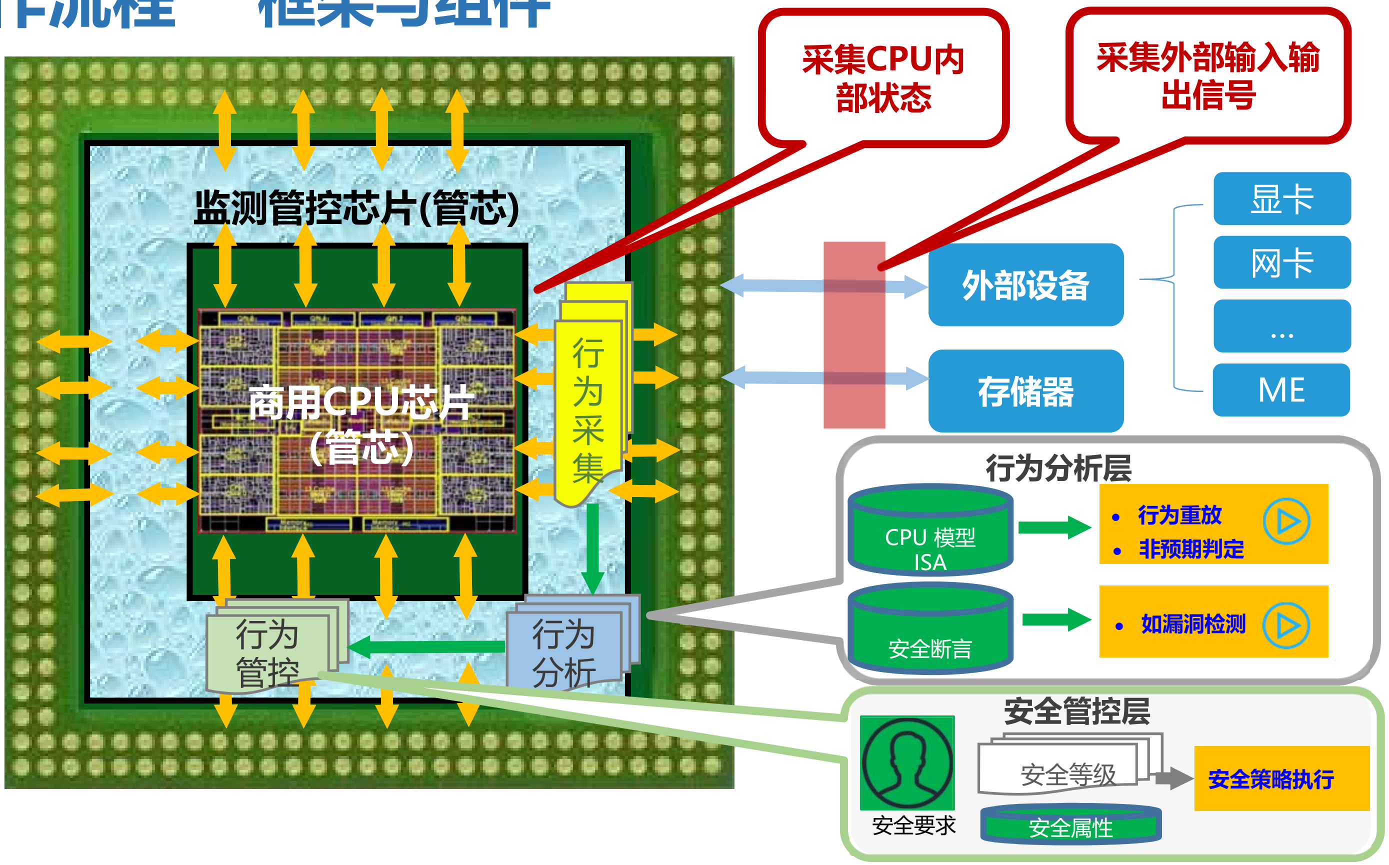
行为管控

- 当行为分析发现与预期不符或者违反安全断言时，与软件手段配合，切断攻击通路

3.2 工作流程 - 框架与组件



3.2 工作流程 - 框架与组件



三、CPU硬件安全动态监测管控技术

3.1 技术原理

3.2 工作流程

3.3 关键技术

3.4 技术验证

3.3 关键技术

行为采集

1、CPU硬件行为非侵入式采集技术与结构化存储方法

2、高速总线上异步事件数据记录与指令边界定序方法

行为分析

3、基于 CPU 安全行为模型样本回放及非预期行为判断方法

4、CPU硬件漏洞攻击的预测回放及检测

行为管控

5、CPU微码管控以及针对ME类子系统的硬件I/O管控技术

6、敏感电路时空动态局部随机重构技术

三、CPU硬件安全动态监测管控技术

3.1 技术原理

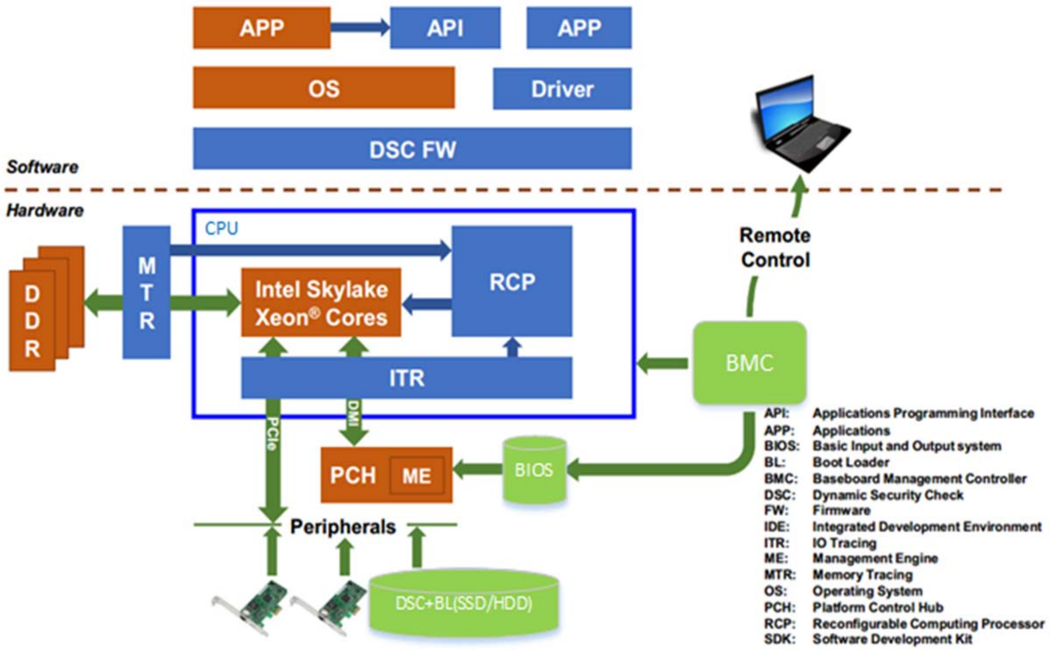
3.2 工作流程

3.3 关键技术

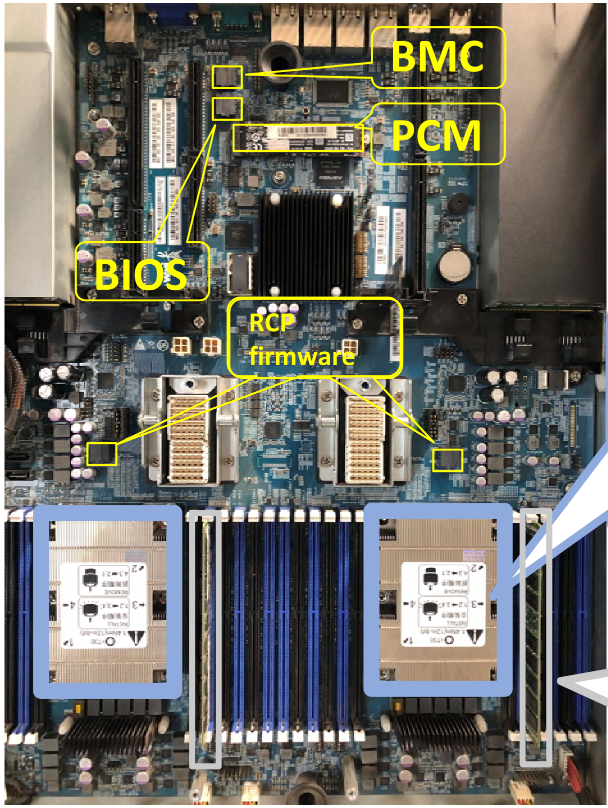
3.4 技术验证

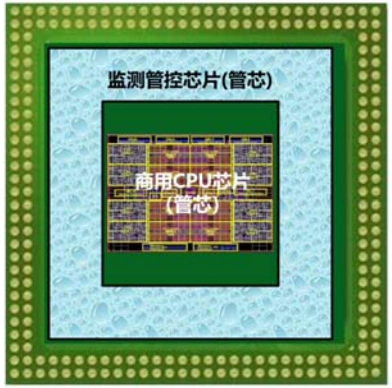
3.4 技术验证 - 实验环境搭建

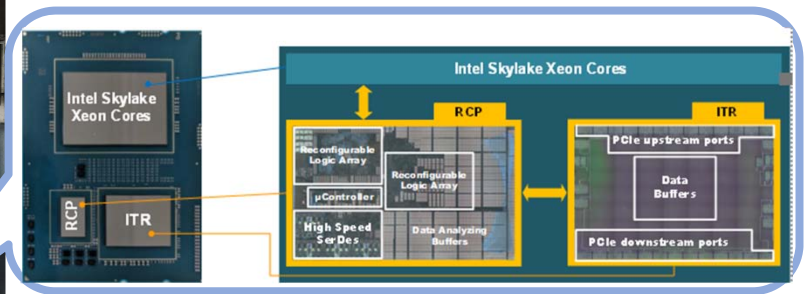
- DSC 原型验证系统：在 Intel 至强处理器上应用DSC技术

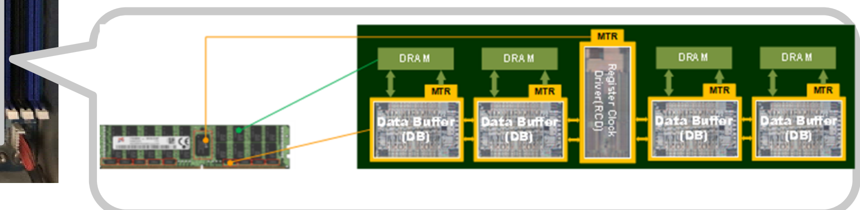


3.4 技术验证 - 实验环境搭建



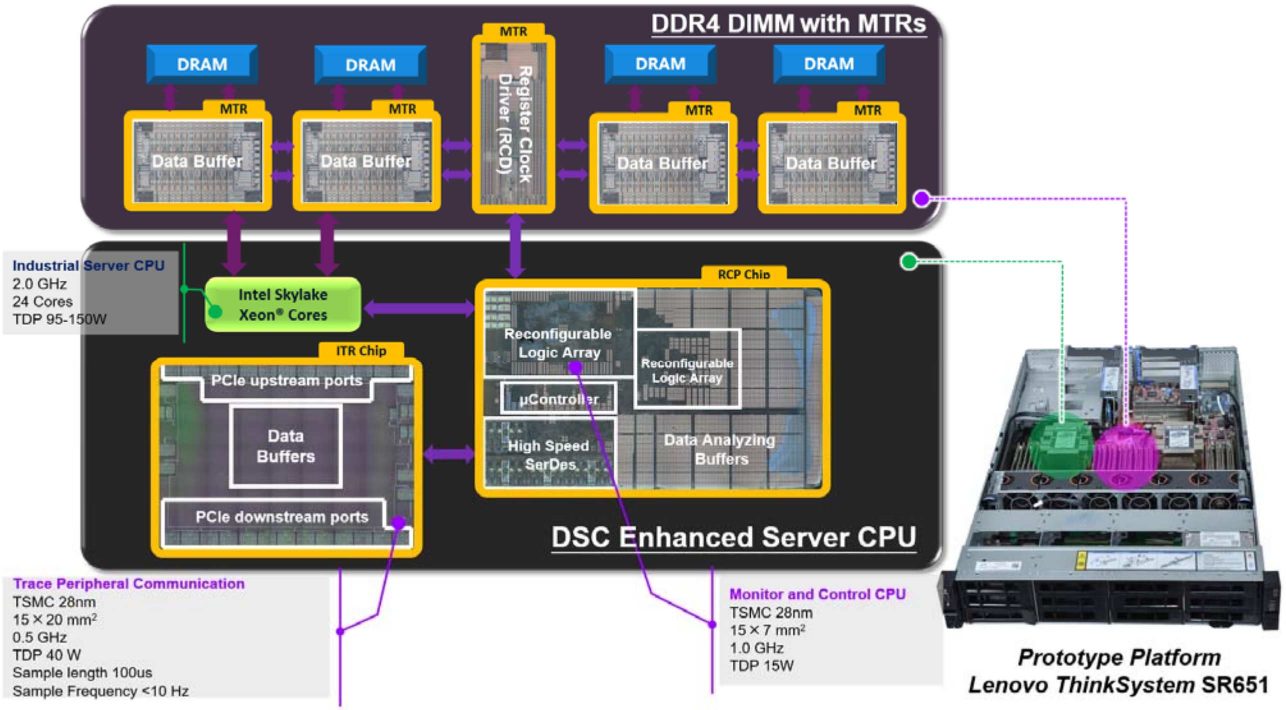






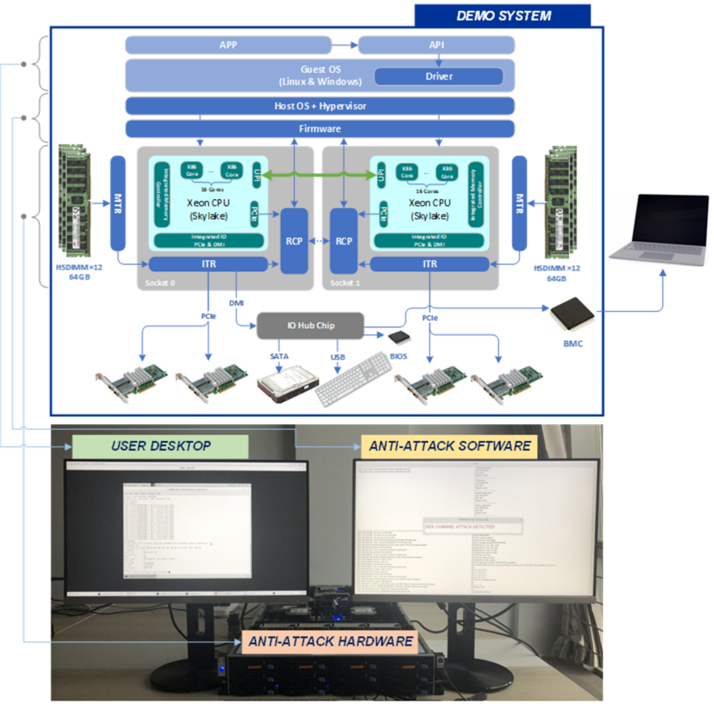
3.4 技术验证 – 实验环境搭建

- DSC原型验证系统



3.4 技术验证 - 实验环境搭建

- 实验环境与方法
  - 用至强处理器验证的理由：
    - 最具有代表性
    - 开发手册开放、完整
    - 不需要开放额外的接口
  - 实验：
    - **硬件后门攻击（硬件木马）**
    - **硬件漏洞攻击**
    - **硬件前门攻击（非法微码更新）**
    - **硬件前门攻击（管理引擎）**

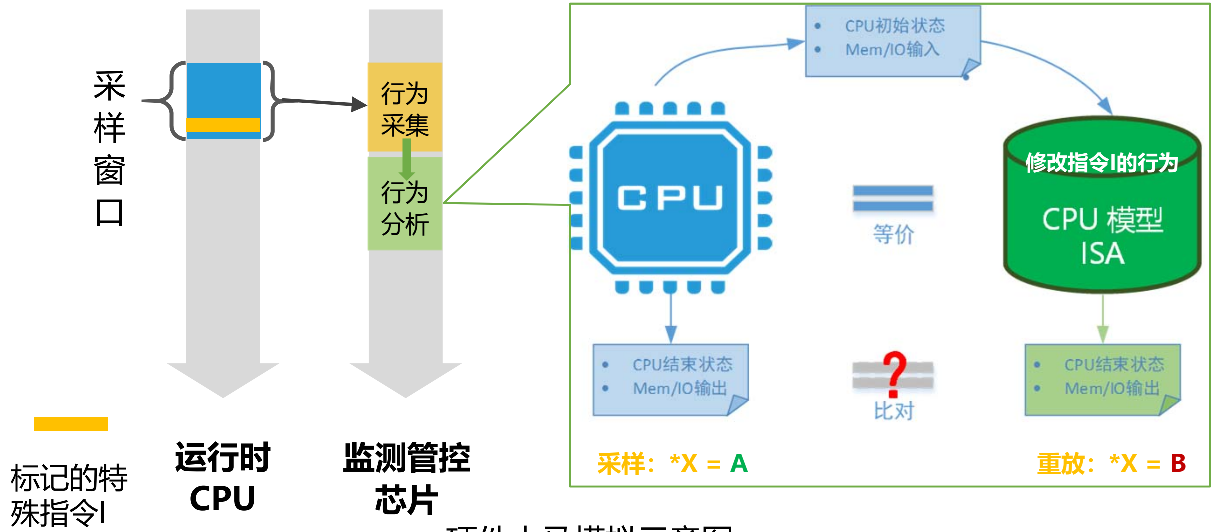


实验环境示意图

3.4 技术验证 - 硬件后门攻击检测（硬件木马）

实验方法：硬件木马模拟

- 硬件木马对外的体现是某条指令行为的非预期性
- CPU上运行一条特定的指令，其输出作为硬件木马
- 通过修改重放模型中特殊指令的行为，使其与真实CPU行为不一致
- 该指令一旦被采集到，经过分析对比就会发现不一致

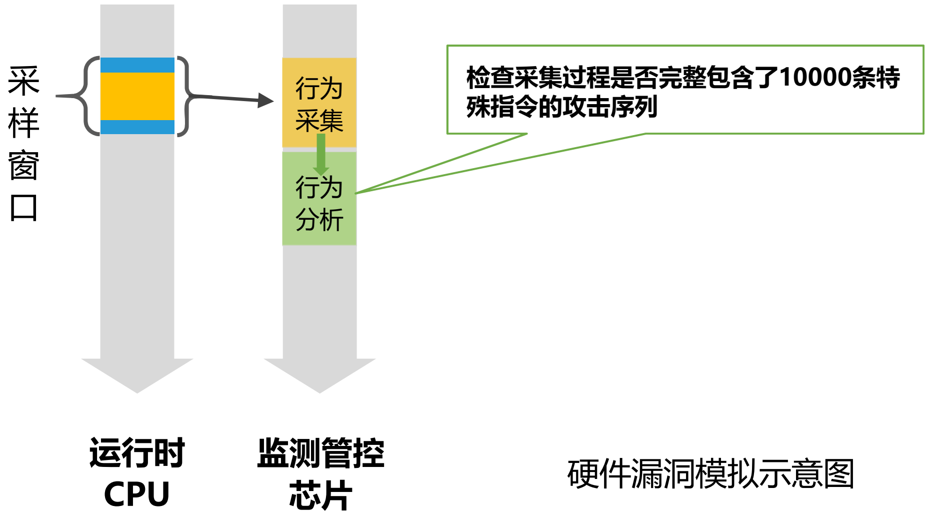


硬件木马模拟示意图

3.4 技术验证 - 硬件漏洞攻击检测

实验方法：硬件漏洞模拟

- 典型的熔断[1]，幽灵[2]每实现一次成功攻击平均需要10000条以上的指令
- 实验简化为对于攻击长度为10000条指令的攻击序列，捕获率为多少



硬件漏洞模拟示意图

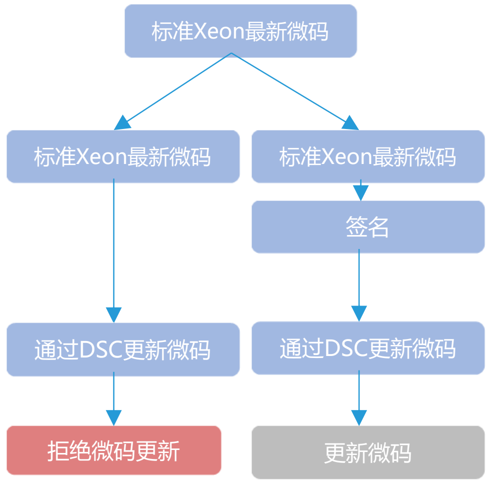
[1] Meltdown, https://meltdownattack.com/

[2] Spectre Attacks: Exploiting Speculative Execution, https://meltdownattack.com/

3.4 技术验证 - 硬件前门攻击抵御（非法微码更新）

- 实验方法：
  - 删除操作系统已有微码
  - BIOS刷为较旧版本（不包含最新微码）
  - 从Intel官方下载Skylake最新微码一份
  - 复制一份微码文件，对其中一份进行签名
  - 使用上述两个微码文件进行微码更新

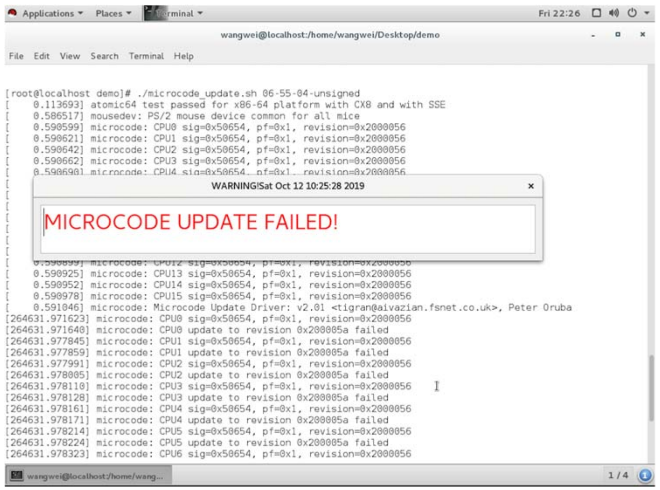
- 预期结果
  - 未经过签名的微码在DSC上被拒绝更新
  - 经过签名的微码被成功更新



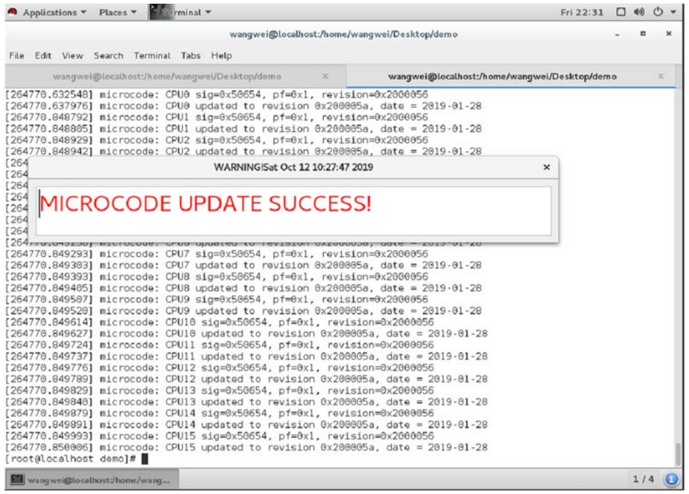
微码更新验证实验原理图

3.4 技术验证 - 硬件前门攻击抵御（非法微码更新）

- 实验结果：
  - DSC平台对未签名的微码拒绝更新
  - DSC平台对经过签名的密码成功更新



未经签名的微码更新失败



经签名的微码更新成功

结论：可100%拒绝未授权的微码更新

提 纲

一、技术背景

二、我国现有应对方案

三、CPU硬件安全动态监测管控技术

四、技术应用探讨

五、总结

四、技术应用探讨

作为我国现有安全体系的补充和拓展

商业部署前

部署于国家权威检测机构中

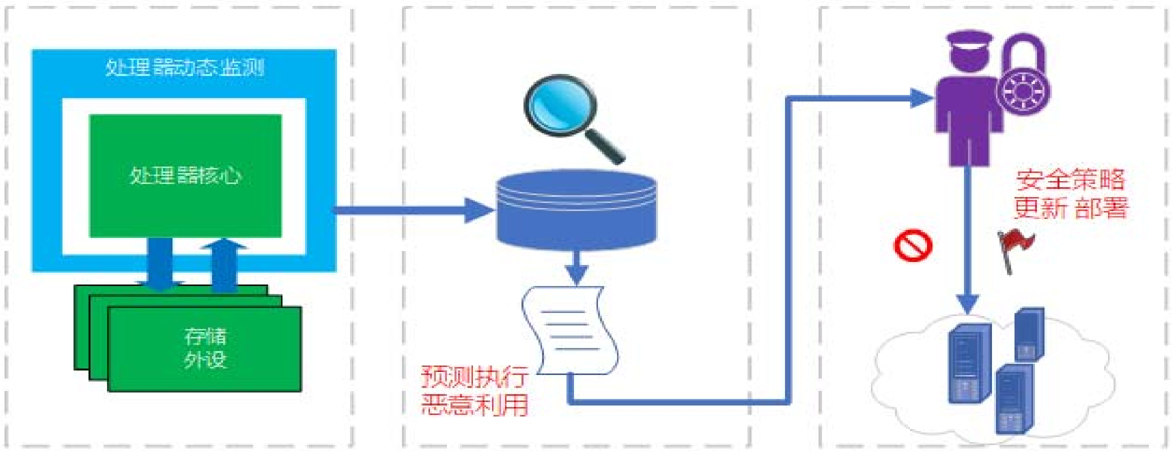
- CPU安全检测平台

商业部署后

部署于开放市场的商用服务器中

- 存量和增量问题均有考虑

四、技术应用探讨 - 部署于开放市场商用服务器中

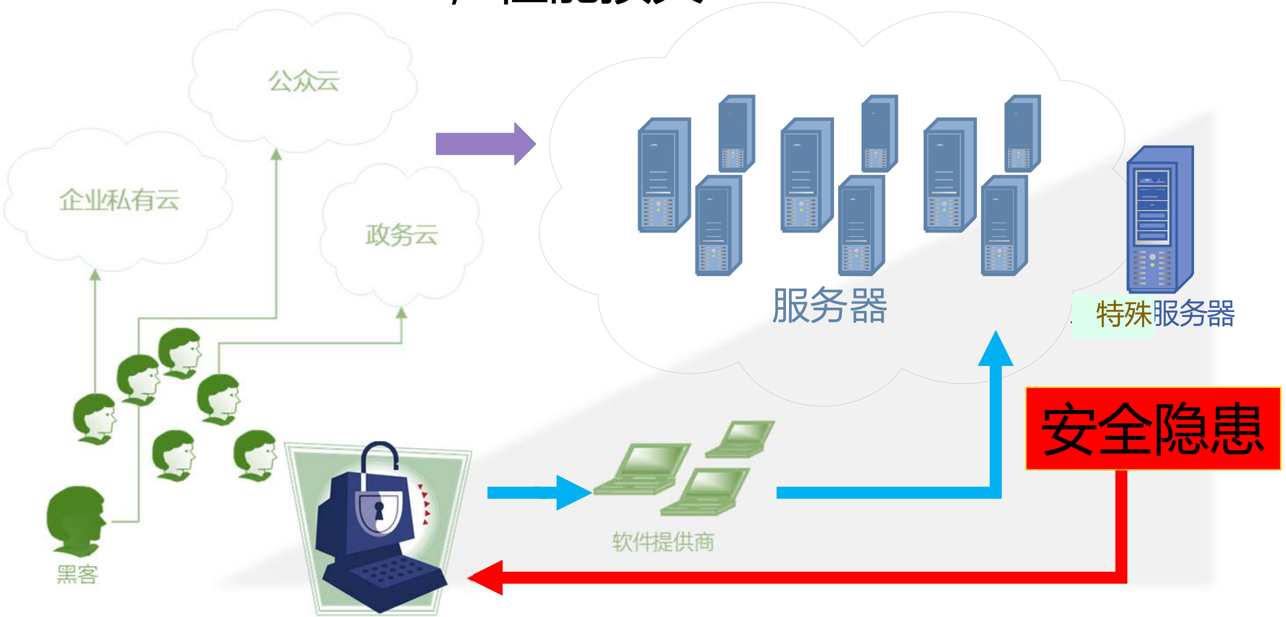


检测 **（软件+硬件）**　　预警（软件）　　抵御（软件）

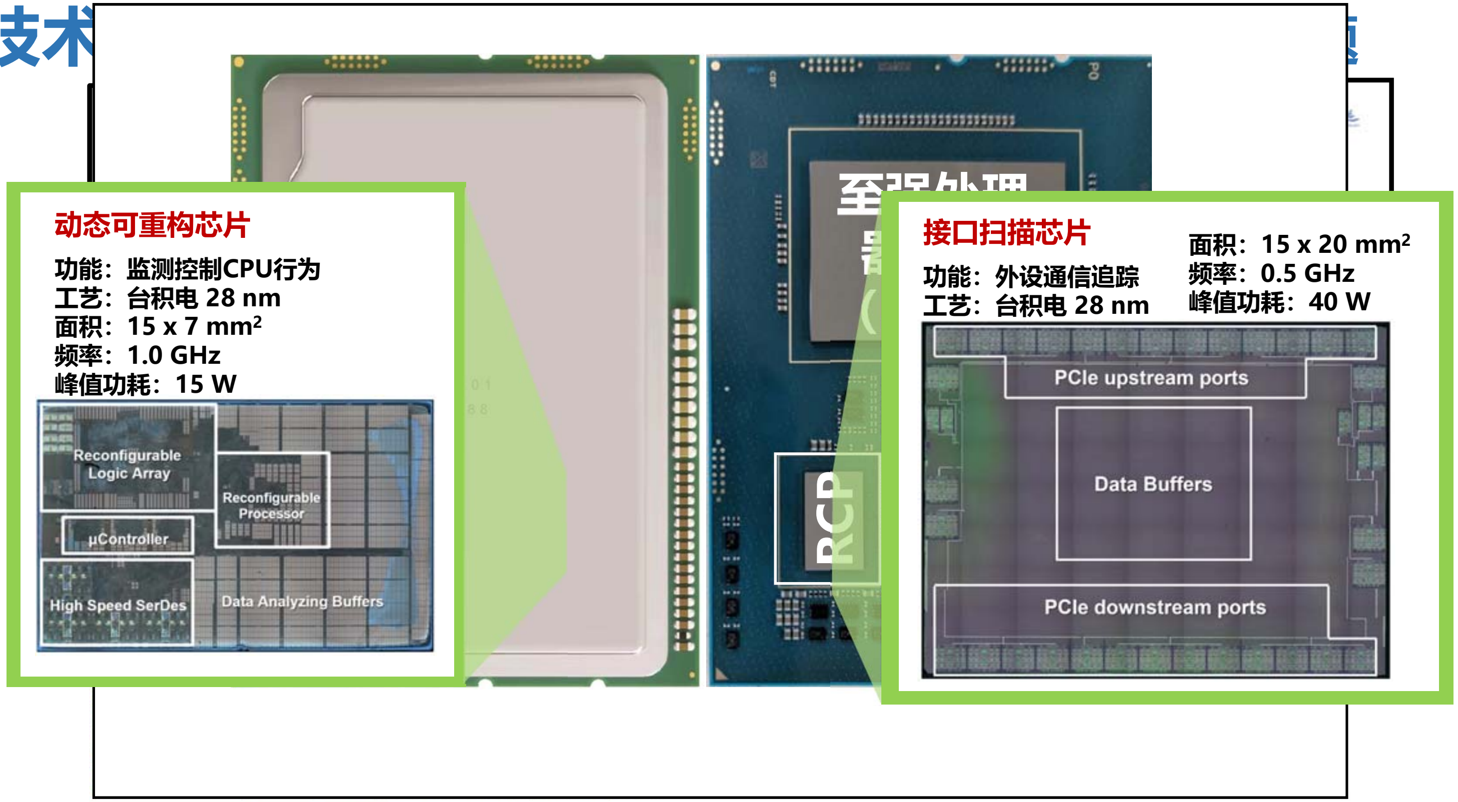
- **建立国家硬件安全问题数据库**
- **构建全国性的硬件风险预警和抵御机制**

四、技术应用探讨 - 解决我国已有1500万台服务器安全问题

- 推算结果：部署30万台特殊服务器，单次攻击检出率99.8%，性能损失0.98%



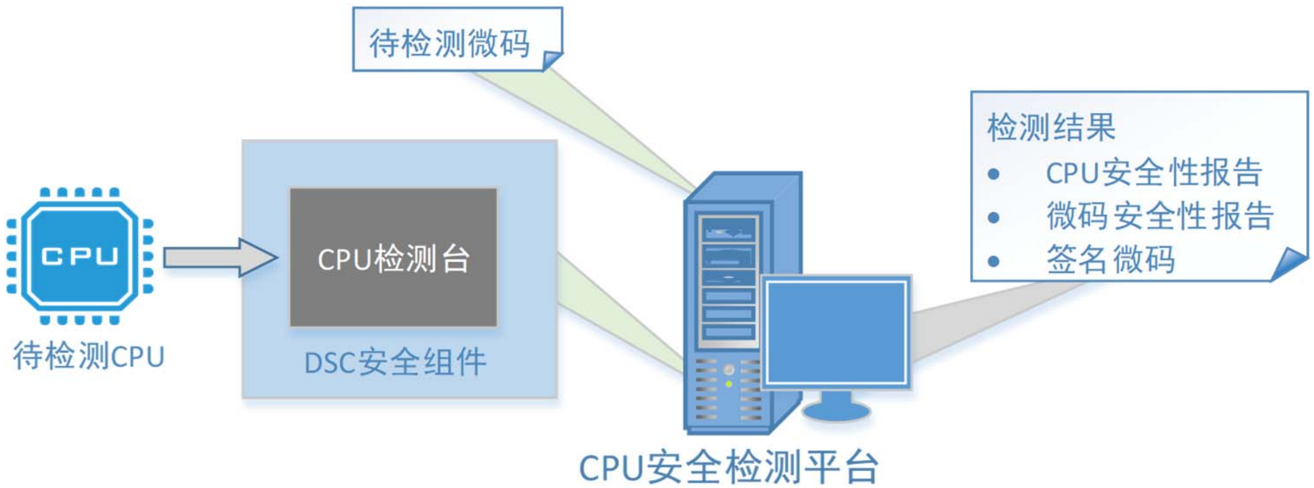
四、技术



高性能商用服务器： 联想 SR651、 长城 GW920、新华三 R4900

四、技术应用探讨 - 研制检测仪器部署于国家权威检测机构

- CPU芯片商业部署前的硬件安全威胁检测
- CPU微码更新前的安全认证、签名发布



某国家权威检测中心

四、技术应用探讨

作为我国现有安全体系的补充和拓展

商业部署前

部署于国家权威检测机构中

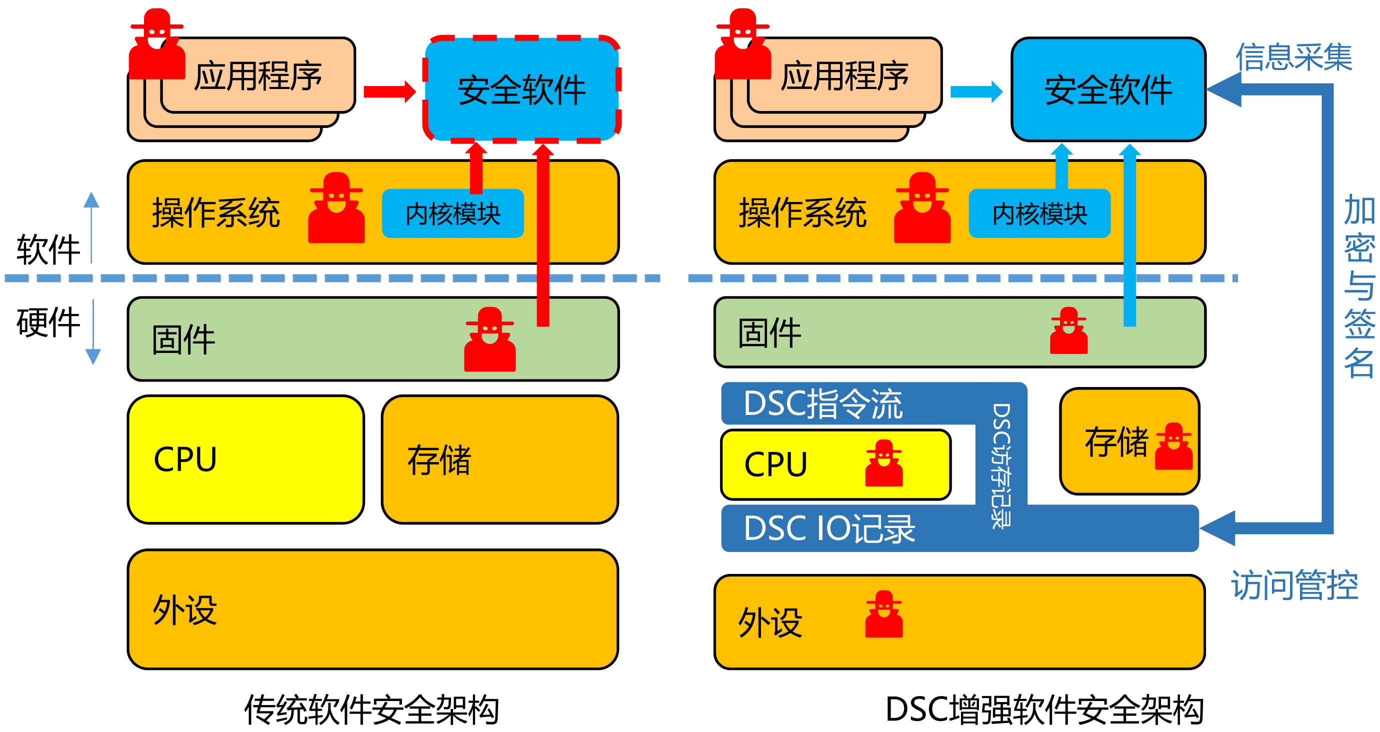
- CPU安全检测平台

商业部署后

部署于开放市场的商用服务器中

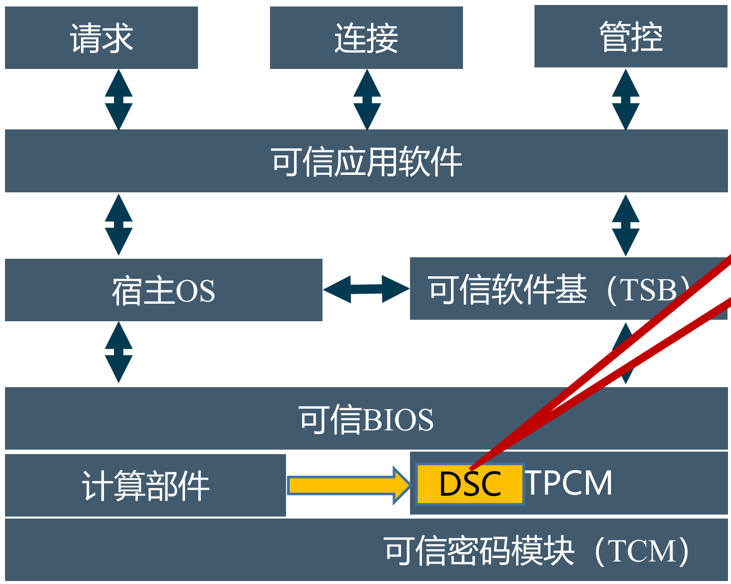
- 存量和增量问题均有考虑

四、技术应用探讨 – 对我国现有安全体系的补充和拓展

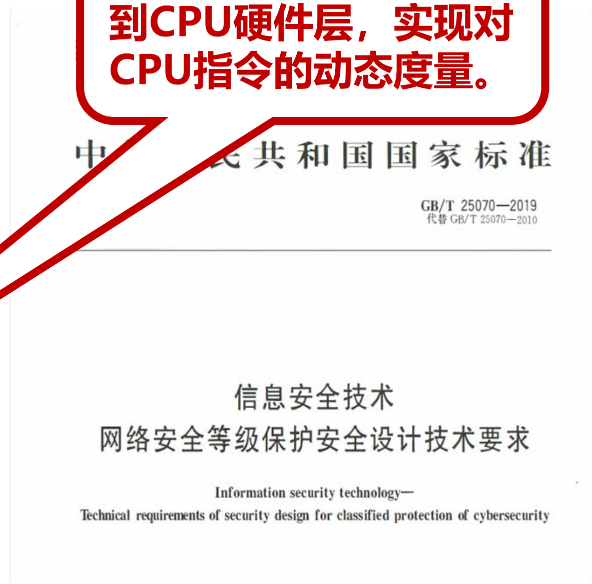


四、技术应用探讨 – 对现有可信计算体系的扩展

将可信计算的度量扩展到CPU硬件层，实现对CPU指令的动态度量。



创新主动免疫体系结构


中华人民共和国国家标准

GB/T 25070—2019
代替 GB/T 25070—2010

信息安全技术
网络安全等级保护安全设计技术要求

Information security technology—
Technical requirements of security design for classified protection of cybersecurity

等保2.0（国家标准）将于2019.12.1实施

[1] 沈昌祥，用可信计算筑牢网络安全防线，2017年报告

四、技术应用探讨 – CPU安全可控国家标准

“必要时可通过技术手段等方式进行评价”……

“通过技术手段等方式进行评价的，评价结果可作为打分依据”……

ICS 35.040
L 80

GB

中华人民共和国国家标准

GB/T 36630.2—2018

信息安全技术
信息技术产品安全可控评价指标
第 2 部分：中央处理器

表 1　中央处理器安全可控评价指标项及指标

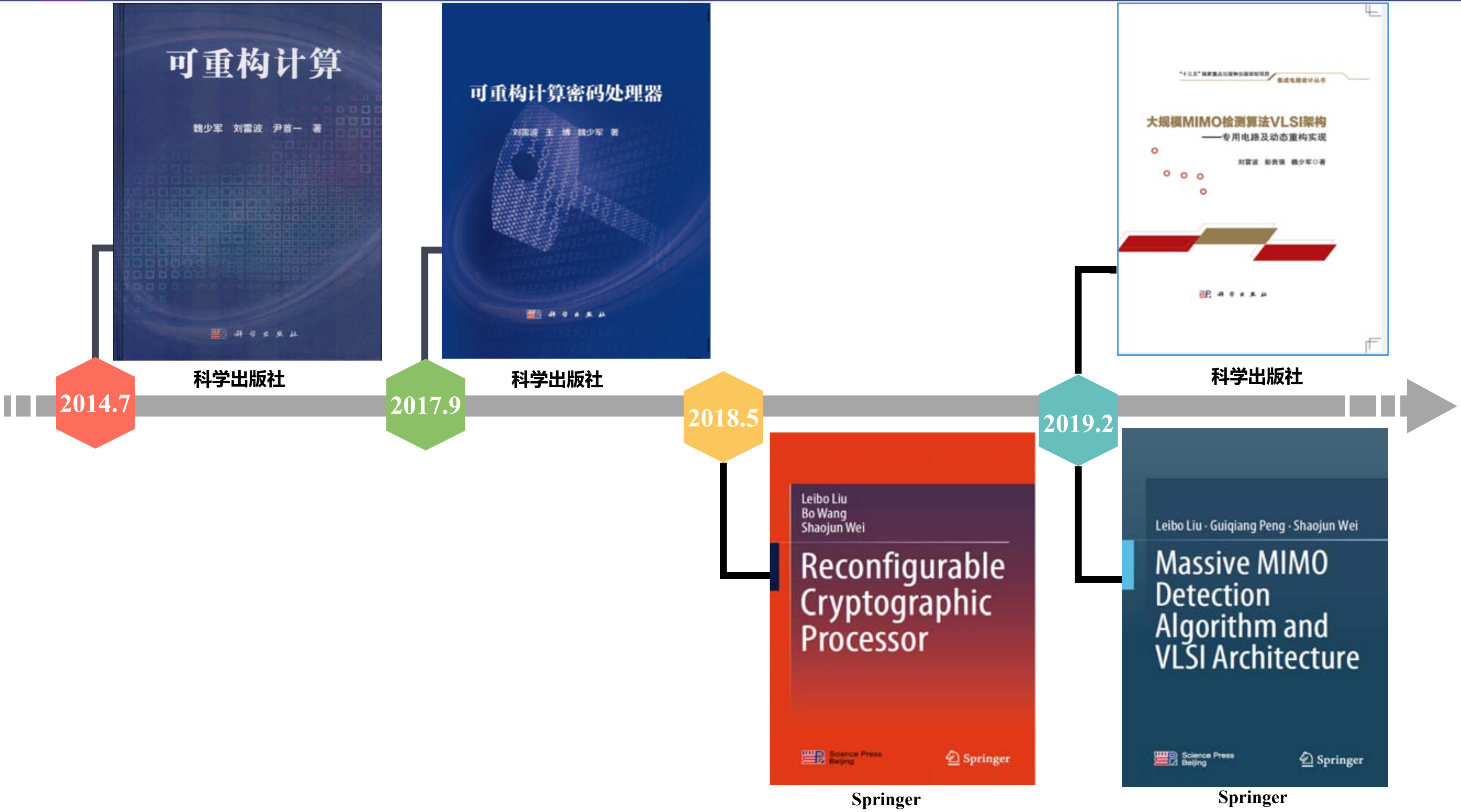
…项	指标说明
…产权状况[a]	根据产品供应方所提供的知识产权拥有情况、获得外… …情况及知识产权纠纷情况等相关材料，对其产品知识产权进行评价
…实现透明性	根据产品供应方所提供的关键技术相关材料的真实性、…核查性、规范性和完备性，对其设计实现透明性进行评价，必要时可通过技术手段等方式进行评价
…设计验证	对产品设计验证环境、设计验证充分性、设计验证结果与产品一致性等进行评价，必要时可通过技术手段等方式进行评价
关键技术…发能力	对产品供应方关键技术掌控能力、基于安全可控需求修改指令集架构及修改或替换关键 IP 核[b] 的权限及能力进行评价
…全生态适应性	对产品所适配操作系统的安全可控程度和密码合规性[c] 进行评价，必要时可通过技术测试的方式辅助评价
…续供应能力	对产品供应方的产品供应情况和核心团队情况进行评价
…应链保障能力	对产品设计环节、流片环节、封装环节、测试环节的供应链保障能力进行评价
…服务保障能力	对产品供应方服务及时性和服务质量进行评价

…中央处理器产品相关知识产权，包括但不限于商标、著作权、专利、集成电路布图。

表 2（续）

指标项	…内容	分值	评分说明
产品服务保障能力	服务及…	4	拥有专业的本地服务团队，能够提供原厂级服务，具备面向全国范围内的产品应用方做出服务响应的能力，能提供及时有效的服务。（4 分） 拥有专业的本地服务团队，具备面向全国范围内的产品应用方做出服务响应的能力，能提供及时有效的服务。（2 分） 没有专业的本地服务团队，或不能提供及时有效的服务。（0 分）
	服务规范	4	有明确的产品服务质量承诺，建立了全面的产品服务体系，能够保证产品服务过程的安全性。（4 分） 没有明确的产品服务质量承诺或有承诺不履行，或没有建立产品服务体系，或产品服务过程存在安全隐患。（0 分）

[a] 通过技术手段等方式进行评价的，评价结果可作为打分依据。
[b] 若设计验证结果与产品不一致，则产品设计实现透明性相关材料不满足真实性要求。


2014.7
可重构计算
魏少军 刘雷波 尹首一 著
科学出版社
2017.9
可重构计算密码处理器
科学出版社
2018.5
Leibo Liu
Bo Wang
Shaojun Wei
Reconfigurable Cryptographic Processor
Springer
2019.2
大规模MIMO检测算法VLSI架构
——专用电路及动态重构实现
科学出版社
Leibo Liu · Guiqiang Peng · Shaojun Wei
Massive MIMO Detection Algorithm and VLSI Architecture
Springer


World Internet Conference
世界互联网领先科技成果发布活动(2018)
RELEASE CEREMONY FOR WORLD LEADING INTERNET SCIENTIFIC AND TECHNOLOGICAL ACHIEVEMENTS (2018)
清 华 大 学
Tsinghua Unversity
CPU硬件安全动态监测管控技术
Dynamic Check and Control Technology for CPU Hardware Security
经第五届“世界互联网大会·乌镇峰会”世界互联网领先科技成果推荐委员会推荐，
作为世界互联网领先科技成果发布，兹以证明。
This is to certify that your achievement, recommended by the Recommendation Committee for World Leading Internet Scientific and Technological Achievements of the 5th World Internet Conference (WIC), is released as the World Leading Internet Scientific and Technological Achievement.
世界互联网大会·乌镇峰会组委会
Organizing Committee of the World Internet Conference (Wuzhen Summit)
2018年11月7日 November 7, 2018


CCTV 2
财经
浙江乌镇 直播
互联网
Internet
能源
Energy
电信
Telecommunications
CPU硬件安全动态监测管控技术
Dynamic Check and Control Technology for CPU Hardware Security
津逮® CPU及其相关软硬件产业
Jintide® CPU and Related Ecosystem
2018世界互联网领先科技成果发布
清华大学发布：CPU硬件安全动态监测管控技术

提 纲

一、技术背景

二、我国现有应对方案

三、CPU硬件安全动态监测管控技术

四、技术应用探讨

五、总结

五、总结

- 芯片硬件安全问题比想象的来得更快、更严重
  - **我国服务器CPU一段时期内依赖进口，硬件安全问题突出**

- 提升芯片硬件安全性：**软件手段+行政手段，效果并不明显**

- 软硬结合是解决CPU硬件全问题的关键手段
  - **"CPU芯片硬件安全动态监控 + 软件技术"是有效手段之一**


XDef 2019
对话 · 交流 · 合作　前沿 · 实用 · 人才
Thanks
谢谢关注！
69